



Escola d'Enginyeria de Telecomunicació i
Aeroespacial de Castelldefels

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

DEGREE FINAL PROJECT

TITLE: Evaluation of Remain Well Clear and Collision Avoidance for Drones

DEGREE: Bachelor's degree in Aerospace Systems Engineering

AUTHOR: Santiago L. del Hierro Mosteiro

DIRECTOR: Enric Pastor Llorens

DATE: October 19th, 2021

Title: Evaluation of Remain Well Clear and Collision Avoidance for Drones

Author: Santiago L. del Hierro Mosteiro

Director: Enric Pastor Llorens

Date: October 19th, 2021

Overview

One of the cornerstones that should enable inserting unmanned aircraft into the airspace is the development of Detect and Avoid (DAA) systems. DAA systems will improve the Remote Pilot (RP) situational awareness by means of electronic conspicuity devices, providing RPs with the necessary means to Remain Well Clear (RWC) from other traffic and, if necessary, avoid Mid-Air collisions (MAC). DAA systems will compensate for the loss of a pilot on board, which drastically reduces the capacity to keep a safe separation from traffic, making current Rules of the Air very challenging to achieve.

Given the growing popularity of drone operations for commercial and recreational purposes, new standards should include them in the not-too-distant future. Since current DAA standards and algorithms (DO-365 and ED-258) are being developed targeting large, mostly military Remotely Piloted Aircraft Systems (RPAS), this project proposes a new set of detection volumes and alert thresholds for U-Space users according to an aircraft type classification. This will allow adapting the existing DAA algorithms to small drones, complying with the new European framework of services and applications for drones (U-Space).

Because testing new safety nets (such as new DAA algorithms) on real aircraft would be dangerous and inadequate, radar reports and computer-based simulations allow for a risk-free and faster evaluation of safety net performances. Due to the current lack of real drone radar tracks, this project has developed a multi-rotor drone encounter generator tool (called DEG). This software is able to generate a large number of synthetic pairwise quadcopter drone conflict tracks, simulating the instant prior to a MAC. The way trajectories are generated by DEG strongly depends on the type of operation being flown (inspection/surveillance flights and logistic flights) and the aircraft type (including a DJI F450 and a faster version called DJI F450 FAST).

The results of this project include a drone conflict trajectory example generated with DEG and an investigation of the performance and effectiveness of the DEG tool using a tailored existing DAA algorithm (DAIDALUS).

Títol: Avaluació de *Remain Well Clear* i *Collision Avoidance* per drons.

Autor: Santiago L. del Hierro Mosteiro

Director: Enric Pastor Llorens

Data: 19 d'Octubre de 2021

Resum

Una de les peces clau que permetrà inserir aeronaus no tripulades a l'espai aeri és el desenvolupament de sistemes de detecció i evasió (DAA). Els sistemes DAA milloraran les capacitats dels pilots remots (RP) de visió del trànsit mitjançant l'ús de dispositius electrònics, proporcionant als RP els mitjans necessaris per mantenir-se ben separat (RWC) d'altres aeronaus i, si cal, evitar una col·lisió a l'aire (MAC). Els sistemes DAA compensaran la pèrdua d'un pilot a bord, cosa que redueix dràsticament la capacitat de mantenir una separació segura amb altres aeronaus, fent que les Regles de l'Aire actuals siguin molt difícils d'aconseguir.

Donada la creixent popularitat de les operacions de drons amb finalitats comercials i recreatives, les noves normes hauran d'incloure'ls en un futur no gaire llunyà. Atès que els estàndards i algorismes DAA actuals (DO-365 i ED-258) s'estan desenvolupant per grans aeronaus pilotades remotament (RPAS), majoritàriament militars, aquest projecte proposa un nou conjunt de volums de detecció i llindars d'alerta per als usuaris del U-Space segons una classificació del tipus d'aeronau. Això permetrà adaptar els algorismes DAA existents a drons petits, complint amb el nou marc europeu de serveis i aplicacions per a drons (U-Space).

Degut a que provar noves xarxes de seguretat (com ara nous algorismes DAA) en aeronaus reals seria perillós i inadequat, les deteccions amb radars i les simulacions basades en ordinador permeten una avaluació sense risc i més ràpida pel càlcul del rendiment d'aquestes xarxes. A causa de la manca actual de trajectòries dron detectades per radars, aquest projecte ha desenvolupat una eina de generació de conflictes per a drons multi rotors (anomenada DEG). Aquest *software* és capaç de generar una gran quantitat de trajectòries conflictives sintètiques entre parelles de drons quadcopter, simulant l'instant previ a una col·lisió aèria. La forma en què DEG genera les trajectòries depenen en gran mesura del tipus d'operació que es fa (vols d'inspecció / vigilància i vols logístics) i del tipus d'aeronau (inclòs un DJI F450 i una versió més ràpida anomenada DJI F450 FAST).

Els resultats d'aquest projecte inclouen un exemple de trajectòria conflictiva de drons generat amb DEG i una investigació del rendiment i l'eficàcia de l'eina DEG mitjançant un algorisme DAA existent (DAIDALUS).

I would like to express my deepest appreciation to the director of the project, Mr. Enric Pastor Llorens, who proposed many of the fields and topics of the research, and who contributed with many innovative ideas. In addition, he gave me the opportunity to experience a 6-month traineeship at EUROCONTROL's Experimental Centre in Brétigny, France.

I extend my gratitude to my parents, who have given me their unconditional support during all these years.

INDEX

INTRODUCTION.....	12
CHAPTER 1. U-SPACE PRINCIPLES AND CONFLICT MANAGEMENT ...	14
1.1 U-Space Volumes and Conflict Resolution.....	14
1.2 U-Space Services.....	15
1.3 Separation Management and Collision Avoidance	16
1.3.1 Manned Aviation	16
1.3.2 Unmanned Aviation	18
1.4 Rules of the Air and Flight Rules	19
1.4.1 EASA – EUROCONTROL: Low Level Flight Rules.....	20
CHAPTER 2. DETECT AND AVOID CONCEPT AND SPECIFICATIONS... 21	
2.1 Remain Well Clear (RWC) Definition.....	21
2.2 Collision Avoidance (CA) Definition	22
2.3 Mathematical Background and Metrics for DAA	22
2.3.1 Time and Distance Metrics	22
2.3.2 Hazard Alert Zone (HAZ) Definition	23
2.3.3 Hazard Zone Violation	24
2.4 Alert Types and Hierarchy	26
2.5 Manoeuvres to Remain Well Clear	27
2.6 DAA Equipment	28
CHAPTER 3. U-SPACE DETECT AND AVOID ALGORITHM	29
3.1 Aircraft Classification.....	29
3.1.1 Class A	29
3.1.2 Class B1 & B2.....	30
3.1.3 Class C	31
3.1.4 Summary Table	32
3.2 RWC & CA Boundaries.....	32
3.2.1 NMAC Volume (NMACv)	33
3.2.2 Well Clear Volume (WCV)	33
3.3 Tracking and Alerting Process.....	36
3.3.1 Well Clear Logic.....	36
3.3.2 Detection Logic	36
3.3.3 Alerting Logic	37
CHAPTER 4. MULTI-ROTOR DRONE ENCOUNTER MODEL.....	39
4.1 Architectural Design.....	39
4.1.1 Generation of Segments.....	40

4.1.2	4D Waypoints	42
4.1.3	Drone Modelling.....	43
4.1.4	Trajectory Simulation.....	47
4.1.5	Intruder's path rotation.....	48
CHAPTER 5. RESULTS.....		49
5.1	DEG: Conflict Trajectory Example	49
5.2	Characterizing Unmitigated DEG Encounters	59
CHAPTER 6. CONCLUSIONS & FURTHER IMPROVEMENTS		61
CHAPTER 7. CLOUD-BASED U-SPACE TACTICAL CONFLICT RESOLUTION SERVICE		62
7.1	Data Filtering	62
7.1.1	Coarse Filter	63
BIBLIOGRAPHY.....		65
APPENDIX.....		67
APPENDIX A. DYNAMIC WELL CLEAR VOLUME DEMONSTRATION.....		68
APPENDIX B. COMPREHENSIVE DESCRIPTION OF DAIDALUS ALGORITHM.....		70
B.1	Well Clear Logic.....	70
B.2	Detection Logic.....	70
APPENDIX C. HORIZONTAL AND VERTICAL DRONE PERFORMANCE TEST		74
C.1	Horizontal and Vertical Performance Test – DJI F450	74
C.2	Horizontal and Vertical Performance Test – DJI F450 FAST.....	80
APPENDIX D. DEG ARCHITECTURAL DESIGN		85

LIST OF FIGURES

Fig 1 CORUS Very Low-Level Airspace Volumes	15
Fig 2 U-Space Services.....	16
Fig 3 Anti-Collision Barriers	16
Fig 4 Air Traffic Management Overview	17
Fig 5 VFR/LFR Boundaries	20
Fig 6 Well Clear Volume and Near Mid-air Collision Volume	23
Fig 7 HAZ violation based on the Modified Tau criterion: Visual Description ...	25
Fig 8 Alerting Process Timeline Concept	27
Fig 9 NASA's DAIDALUS Horizontal Manoeuvre Guidance Concept.....	27
Fig 10 Example of LIDAR, electro-optical and ultrasonic sensors	28
Fig 11 DEG Encounter Generation Procedure	40
Fig 12 CPA (or NMAC) Segment Generation Concept.....	42
Fig 13 120 Degree heading change in steps of 45 degrees.	43
Fig 14 DJI F450 Quadcopter.	43
Fig 15 Translational and Rotational Movements of a 6-DOF Model.....	45
Fig 16 PID Close-loop Feedback Illustration	47
Fig 17 Ownship Desired Trajectory: Horizontal View	50
Fig 18 Ownship Actual Flight Path: Horizontal View.....	50
Fig 19 Ownship Desired Trajectory: Speed vs Time	51
Fig 20 Ownship Actual Flight Path: Speed vs Time.....	51
Fig 21 Ownship Desired Trajectory: Altitude vs Time.....	52
Fig 22 Ownship Actual Flight Path: Altitude vs Time	52
Fig 23 Intruder Desired Trajectory: Horizontal View	53
Fig 24 Intruder Actual Flight Path: Horizontal View	53
Fig 25 Intruder Desired Trajectory: Speed vs Time	54
Fig 26 Intruder Actual Flight Path: Speed vs Time	54
Fig 27 Intruder Desired Trajectory: Altitude vs Time	55
Fig 28 Intruder Actual Flight Path: Altitude vs Time.....	55
Fig 29 DEG Encounter: Horizontal View	56
Fig 30 DEG Rotated Encounter: Horizontal View	56
Fig 31 DEG Encounter: Speed vs Time	57
Fig 32 DEG Rotated Encounter: Speed vs Time	57
Fig 33 DEG Encounter: Altitude vs Time	58
Fig 34 DEG Rotated Encounter: Altitude vs Time	58
Fig 35 Proposed Cloud-Based Server system overview for autonomous flight operations	62
Fig 36 Cloud-based System Processing Overview.....	63
Fig 37 Conflict status according to intruder position	68
Fig 38 Generated heatmap plot examples	69
Fig 39 Horizontal and Vertical Performance Test Scenario.....	74
Fig 40 Horizontal and Vertical Performance Test Scenario.....	80

LIST OF TABLES

Table 2.1 Alert Descriptions and Hierarchy	26
Table 3.1 Class A performance parameters.....	30
Table 3.2 Class B1 and B2 performance parameters.....	30
Table 3.3 Class C performance parameters.....	31
Table 3.4 : General Performance Parameters by Class.	32
Table 3.5 Proposed NMAC Volume parameters.	33
Table 3.6 DAA system processing time and pilot reaction time.....	34
Table 3.7 Proposed TAUMOD and TCOA values.	34
Table 3.8 CORUS, DO-365 and MIT WCV thresholds.....	35
Table 3.9 Proposed DMOD and ZTHR parameters.....	36
Table 3.10 Proposed Alert Time Thresholds.....	37
Table 4.1 Steps for Generating the NMAC Segment.....	40
Table 4.2 Steps for Generating Forwards and Backwards Segments.....	41
Table 4.3 4D Waypoints Calculation	43
Table 4.4 Drone Limits and Operational Envelope.	44
Table 4.5 Differences in Performance Characteristics Between Models.	44
Table 4.6 : Aircraft Dynamic State Variables.....	46
Table 4.7 Motor Dynamics State Variables	46
Table 5.1 DEG Generated Encounter: Initial CPA Conditions.....	49
Table 0.1 Architectural Design of DEG.....	85
Table 0.2 DEG Component Descriptions.	86

ACRONYMS, ABBREVIATIONS AND DEFINITIONS

AAV	Autonomous passenger Air Vehicles
ACAS	Airborne Collision Avoidance Systems
ADS-B	Automatic Dependent Surveillance Broadcast
AGL	Above Ground Level
ATAR	Air To Air Radar
ATCO	Air Traffic Control Operators
ATS	Air Traffic Services
BER	Bit Error Rate
BVLOS	Beyond Visual Line of Sight
CA	Collision Avoidance
CoC	Clear of Conflict
CORUS	Concept of Operations for European U-space Services
CPA	Closest Point of Approach
CS	Control Station
DAA	Detect and Avoid
DAIDALUS	Detect and Avoid Alerting Logic for Unmanned Systems
DEG	Drone Encounter Generator
DMOD	Distance Modification
DOF	Degree of Freedom
DWC	DAA well clear
EASA	European Union Aviation Safety Agency
ENU	East-North-Up
EO	Electro-Optical
EUROCAE	European Civil Aviation Equipment Organization
FAA	Federal Aviation Administration
FTS	Fast Time Simulation
GA	General Aviation
GNSS	Global Navigation Satellite Systems
HAZ	Hazard Alert Zone
HITL	Human-In-The-Loop
HMD	Horizontal Miss Distance
IR	Infrared
IFR	Instrumental Flight Rules
LFR	Low-Level Flight Rules
LIDAR	Laser Identification Detection and Ranging
LoWC	Loss of Well Clear
MAC	Mid-Air Collision
MIT-LL	Massachusetts Institute of Technology – Lincoln Labs
MMW	Microwave Radars
MOPS	Minimum Operational Performance Standards
MTOM	Maximum Take-Off Mass
NASA	National Aeronautics and Space Administration
NED	North-East-Down
NHZ	Non-Hazard Zone
NMACv	NMAC Volume
RA	Resolution Advisory

ROT	Rate of Turn
RoW	Right of Way
RP	Remote Pilot
RPA	Remotely Piloted Aircraft
RTS	Real Time Simulations
RWC	Remain Well Clear
SDS	Spatial Data Structure
SESAR JU	Single European Sky ATM Research Joint Undertaking
SSV	Self-Separation Volume
TCAS	Traffic alert and Collision Avoidance System
TCOA	Time to Co-Altitude
UAV	Unmanned Aerial Vehicles
UTM	Unmanned Traffic Management system
VFR	Visual Flight Rules
VLL	Very Low-level
VLOS	Visual Line of Sight
VMD	Vertical Miss Distance
VTOL	Vertical Take-off Landing
WC	Well Clear
WCV	Well Clear Volume
ZTHR	Vertical Separation Threshold

INTRODUCTION

In recent years there has been a significant increase in Unmanned Aircraft System (UAS) operations for commercial purposes. This is mainly due to advancements in state-of-the-art technology improvements and a clearer regulatory framework, both of which aim to create a market of drone services, supporting job creation and growth in this emerging sector of the economy. It is estimated that by 2035 the drone sector in Europe will have an economic impact of about EUR 10 billion per year, employing more than 100,000 people, mainly in services [1].

One of the cornerstones that should enable inserting unmanned aircraft into the airspace is the development of Detect and Avoid (DAA) systems. DAA systems will improve the Remote Pilot (RP) situational awareness by means of electronic conspicuity devices, providing RPs with the necessary means to Remain Well Clear (RWC) from other traffic and, if necessary, avoid Mid-Air collisions (MAC).

Great efforts are being carried out in both the United States of America (USA) and Europe to build the algorithms and establish the Minimum Operational Performance Standards (MOPS) of DAA systems, in particular but not limited to large military Remotely Piloted Aircraft System (RPAS) operating in controlled airspace (DO-365 and ED-258). Given the growing popularity of drone (or small UAS (sUAS)) operations for commercial and recreational purposes, new MOPS should include them in the not-too-distant future. Evidently, sharing current standards with drones could become extremely inefficient, due to extensive differences in performance, dynamics and operational characteristics.

This project has as first objective to quantify the minimum distances that will ensure U-Space users to stay “well-clear” from other traffic, by proposing a new set of detection volumes and alert thresholds according to an aircraft type classification. These new boundaries will ensure the safeness and maximize the efficiency of any kind of aircraft operation in the European Unmanned Traffic Management (UTM) concept called U-Space.

Every new safety net (such as new DAA systems) must be tested and evaluated before being deployed, to ensure its effectiveness in real hazardous scenarios. Because testing new safety nets on real aircraft would be dangerous and inadequate, radar reports and computer-based simulations allow for a risk-free and faster evaluation of safety net performances. Encounter modelling is a well-established technique for simulating the trajectories prior to a collision (called encounters) between two or more aircraft. Essentially, it allows the generation a large number of artificial but realistic encounters, which statistically represent real operational situations. The safety nets can then be subjected to these encounters in exercises called Fast-Time Simulations (FTS) to test the effectiveness of such systems in simulated environments.

Nowadays, surveillance radars do not provide drone radar track information. Consequently, the only way to validate new DAA systems for drones is by testing them with simulated encounters. As a result, the second objective of this project is to design and develop a new encounter model (called DEG) for small multi-

rotor drones. This tool will produce a large number of pairwise drone conflict trajectories, representing the last moments prior to a MAC. The way trajectories are generated by DEG strongly depends on the type of operation being flown (inspection/surveillance flights and logistic flights) and the aircraft type (including a DJI F450 and a faster version called DJI F450 FAST).

The results of this project include an example of a drone conflict trajectory generated with DEG and an investigation of the performance and effectiveness of the DEG tool using an existing DAA algorithm (DAIDALUS).

Chapter 1 introduces the U-Space, its services, design and the necessary requirements in terms of conflict management. Chapter 2 explains in detail what a DAA system is and what abilities it has. Chapter 3 presents the new set of DAA metrics to adapt NASA's DAA algorithm for large RPAS (called DAIDALUS) to the U-Space environment. Chapter 4 specifies the software architecture, the drone models and the generation of encounters of the new DEG tool. Chapter 5 shows the results of the project. Chapter 6 includes the conclusions and further improvements of the project. Chapter 7 suggests a new way to use cloud servers to provide U-Space tactical drone separation services in highly automated environments and explains the advantages and disadvantages of it.

CHAPTER 1. U-SPACE PRINCIPLES AND CONFLICT MANAGEMENT

The European Commission mandated the Single European Sky ATM Research Joint Undertaking (SESAR JU) to lead the development of an UTM concept for Europe, called U-Space. This UTM system would ensure the safe operation of a large number of drones in low-altitude environments, particularly above urban areas. A blueprint was released in June 2017 with a preliminary vision for the U-space.

U-Space is defined as a new set of services and specific procedures that will support the safe, efficient and secure access to airspace of a large number of drones, relying on a high degree of digitization and automatization. Some of the main objectives of the U-Space include facilitating high-density operations in environments where automated drones are monitored; providing equal access for all users to this airspace minimizing operational costs, taking advantage of current aeronautical services and infrastructure, including satellite navigation systems and communications. The U-Space is intended to be full secure and environmentally friendly, always protecting the privacy of people and their associated data.

1.1 U-Space Volumes and Conflict Resolution

U-Space should not be considered as a volume but as a framework that ensures all kinds of drone operations and categories in all operating environments and in all types of airspace, in particular but not limited to the Very Low-level (VLL) airspace [2]. Since the majority of drone operations will happen in this lowest portion of airspace, this project concentrates on the VLL airspace. This is the airspace below that used by Visual Flight Rules (VFR), under the minimum safe altitude. As amended by the European Commission Implementing Regulation No 923/2012 [3], the minimum height of VFR flights is 1000ft Above Ground Level (AGL) over urban areas and up to 500ft AGL elsewhere. The VLL airspace will be crowded not only with drones but also with police helicopters, armed forces, balloons, gliders, trainings, fire-fighting, ultra-light aircraft, etc.

U-Space divides the VLL airspace into three types of volumes (see [Fig 1](#)), depending on the amount of drone flights expected, the air and ground risks, and the provision of conflict resolution services:

- X Volume: There will be very low traffic demand in this volume. Thus, very low air and ground risk is expected. Only a few services will be offered in this region, excluding any conflict resolution service.
- Y Volume: Aircraft operators must submit an operational plan for approval before flying in airspace Y. Here, conflicts are resolved in the pre-flight phase (strategic conflict resolution). As the risk of collision will be extremely decreased prior to the flight, there will be no tactical (in flight)

conflict resolution service. Only traffic information services will enhance the situational awareness of RPs.

- Z Volume: This volume will be designed to cope with a very high demand of drone operations. Aircraft operating in volume Z must submit an operational plan prior to the flight and must carry a minimum set of electronic equipment onboard. Depending on the provision of tactical conflict resolution services, the Z volume will be splitted into two volumes: (1) Za for controlled airspace where conventional ATS will be in control and will provide services. The use of U-Space services will be optional; (2) Zu which depending on the regulator will be created in uncontrolled airspace with U-Space tactical conflict resolution services providing advice or in controlled airspace in which the aforementioned service will be considered as a conventional ATC service.



Fig 1 CORUS Very Low-Level Airspace Volumes

1.2 U-Space Services

The Blueprint [4] proposes the creation of new U-Space services which will be deployed in an incremental manner. Each new step will propose a new set of services while including an enhanced version of the services already existing (see [Fig 2](#)):

- U1: Foundation services for U-space, including e-registration, e-identification, and geofencing. This will allow inserting and monitoring U-Space users in low density areas.
- U2: U-space initial services for drone operations management, including flight planning, flight approval, tracking, and interfacing with conventional air traffic control.
- U3: Advanced U-space services that support more complicated operations in dense areas, such as conflict detection assistance and automated DAA capabilities.

- U4: U-space full services, providing extremely high degrees of automation, communication, and digitalization for both the drone and the U-space system.

Over time, U-space services will evolve to enable more advanced and complex air operations, increasing the level of automation of drones. All this leads to the development of new conflict mitigation methods based on more advanced electronic conspicuity devices and automated systems.

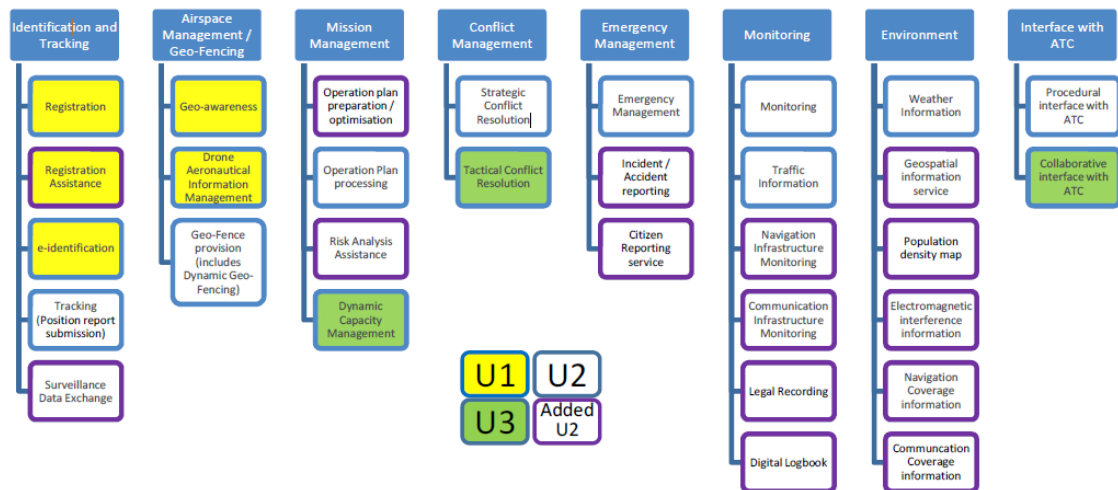


Fig 2 U-Space Services

1.3 Separation Management and Collision Avoidance

One of the major technical challenges of managing air traffic is how to resolve conflicts between aircraft. A "conflict" is defined as a circumstance in which two or more aircraft approximate (or are expected to approximate) each other less than a minimum distance mandated by regulation, generating a hazardous situation.

1.3.1 Manned Aviation

Conflicts in manned aviation are managed by a sequence of anti-collision barriers, each of which is designed to prevent ever-more serious hazards. Depending on the airspace class, the type of flight rules, the phase of the flight and the level of threat different barriers exist (see Fig 3):

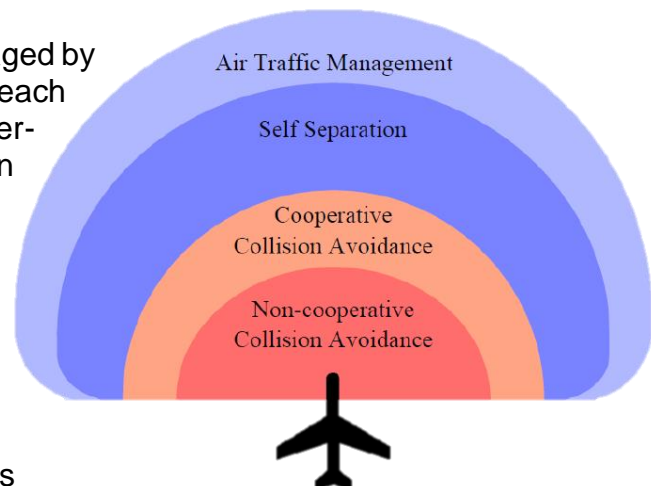


Fig 3 Anti-Collision Barriers

The Air Traffic Management (ATM) applies for any type of aircraft and flight rules (Instrumental Flight Rules (IFR) or VFR) and for any airspace class. ATM is divided into three main services (see [Fig 4](#)):

- Air Traffic Services (ATS): in charge of detecting and preventing any conflict throughout the flight, as well as advising pilots on possible avoidance manoeuvres, thus **providing tactical conflict resolutions**. ATS are composed by Air Traffic Control (ATC), Flight Information Services (FIS) and Alert Services (ALS). Particularly, ATC is constrained by the number of Air Traffic Control Operators (ATCO) available. One of the solutions aimed at reducing the need for ATCOs are automated Conflict Detection and Resolution (CD&R) mechanisms such as the Medium-Term Conflict Detection (MTCD) or the Short-Term Conflict Alert (STCA).
- Air Traffic flow Management (ATFM): trying to balance the traffic demand with the ATC capacity. This service is usually understood as an additional service to ATS aiming at improving the safety, throughput and efficiency. Among others, the ATFM service is in charge of **providing strategic (pre-flight) conflict resolutions**.
- Airspace Management (ASM): targeting the design of the overall airspace by developing ATS routes and Terminal Manoeuvre Area (TMA) procedures, design and implementation of ATS sectorizations, designation of airspace types and coordination between civil/military operations.

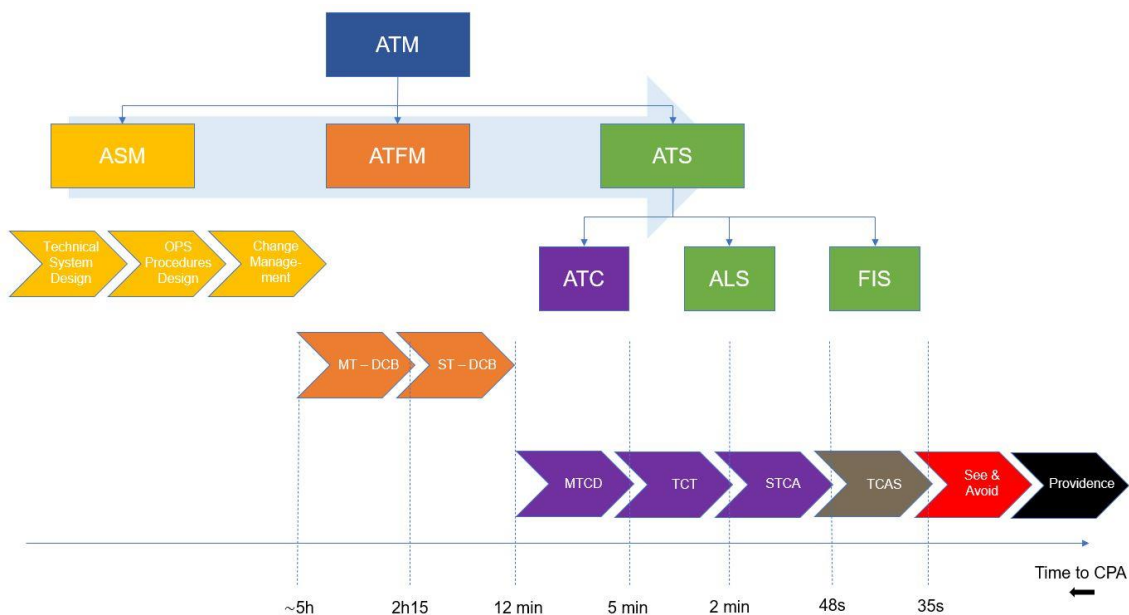


Fig 4 Air Traffic Management Overview

Airborne Separation Assurance Systems (ASAS) may support pilots to conduct more “Free Flights”, allowing them to maintain a self-separation from nearby traffic by increasing the level of situational awareness. Nowadays, ASAS applications might be understood as a supplement of current ATS services. In the long term, and under the right circumstances, suitably equipped aircraft may be able to fly with greater autonomy while also self-separating from other aircraft. It

will be necessary to describe the ATS that will be supplied in this new class of controlled airspace [5]. As technology evolves, new ATM concepts will emerge, enabling the flexible use of airspace by means of user-preferred trajectories, improving airspace capacity, environmental impact, flight efficiency and safety while reducing congestion and costs.

In order to reduce the risk of a MAC, the Airborne Collision Avoidance System (ACAS) was developed. ACAS serves as a last resort safety net, triggering when all previous safety layers have failed to maintain separation. ACAS operates independently of ground-based equipment and advises pilots about probable collisions with surrounding aircraft. Secondary Surveillance Radar (SSR) Mode S transponder-equipped aircraft, known as cooperative aircraft, use ACAS to avoid collisions by interrogating each other, issuing the pertinent alerts and suggesting safety dodging manoeuvres for both aircraft (or even self-executed by the aircraft autopilots). In opposite, conflicts between non-transponder equipped aircraft, commonly associated with non-cooperative aircraft flying VFR in uncontrolled airspace, rely on the “See and Avoid” capability to prevent approaching to surrounding aircraft in such proximity as to generate a collision hazard.

1.3.2 Unmanned Aviation

Fundamentally, the same anti-collision barriers (see [Fig 3](#)) will apply for unmanned aviation. DAA systems will play a major role in the separation management and collision avoidance of UAS. Here the DAA RWC functionality will act as an ASAS, enhancing the RP's situational awareness and providing safe manoeuvres in case of Loss of Well Clear (LoWC), thus allowing self-separation. On the other hand, the DAA CA functionality will equip UAS with same capabilities as ACAS to avoid MAC.

Different DAA systems may be developed depending on the equipment required to fly in each airspace (transponder-equipped or not equipped). For instance, if a RPAS flying in controlled airspace approaches a Mode S transponder-equipped aircraft, then the DAA CA must coordinate with the intruder's ACAS. Otherwise, the DAA CA detection and alerting thresholds shall change to cope with non-cooperative intruders. Moreover, for RPAS flying in controlled airspace the RWC functionality should not override the current separation standards used by ATCOs, usually associated with the boundaries of the STCA. In the case the DAA RWC function suggests a dodging manoeuvre, the RP should coordinate with ATC.

Within the VLL airspace, the procedures become more complex. The U-Space Concept of Operations foreseen by CORUS indicates that in airspace volume Z, tactical conflict resolution services will provide users the ability to RWC by issuing advices and/or instructions to the RPs. Therefore, DAA systems will act as a backup in case this service fails. Only the CA will be implemented by DAA systems. In Y and Z airspace volumes, traffic information will enhance the RP's situational awareness and strategic conflict resolution services will separate trajectories before flight. In Z, all aircraft should be requested to carry a minimum set of DAA equipment onboard to be detectable. For airspace volumes X and Y,

DAA systems (if carried by any user) will be equipped with passive surveillance to search for non-cooperative intruders. How DAA systems will be employed by automated systems is out this project's scope.

1.4 Rules of the Air and Flight Rules

The Rules of the Air are a set of regulations governing matters of air traffic (i.e., meeting and overtaking of other aircraft, the responsibilities of the pilot, the use of defined airways, the minimum height required for flying over cities, etc.) including general rules, visual flight rules and instrument flight rules. In Europe, the Standardised Rules of the Air (SERA) [6] were mandated by the European Commission and developed by EUROCONTROL and the European Aviation Safety Agency (EASA) in 2012.

To date, neither ICAO's Rules of the Air (ICAO Annex 2 [7]) nor SERA Section 3 support UAS operations. These rules do not integrate an exact definition for what it means to be "well-clear" with other aircraft, nor has this minimum safety distance been quantified yet. Furthermore, the lack of a pilot on board not only deteriorates the ability to maintain a safe separation with the surrounding aircraft, but also makes applying the current rules of the air extremely difficult while flying (especially in Beyond Visual Line of Sight (BVLOS) operations). In addition, such rules will not apply for drones in the VLL airspace due to their radically different size, performance, and types of operations when compared to conventional aircraft. Evidently, current regulations do not support any kind of unmanned aircraft operation, thus new Rules of the Air should be established (see Section [1.4.1](#)).

Of utmost importance for the avoidance of collisions are the Right of Way (RoW) rules. These rules are a set of simple guidelines to prioritize certain aircraft and indicate the manoeuvre to follow in case of conflict. As discussed by EUROCONTROL in [8] it is very difficult to give priority to a given aircraft based on the type of operation (Visual Line of Sight (VLOS) or BVLOS) or by the fact that an aircraft is manned or unmanned. In principle, the RoW rules of SERA should remain unchanged but an additional section for VLOS and BVLOS operations, and also with regard to VFR traffic may be included. Perhaps the surveillance to detect possible collisions of very lightweight drones could be exercised outside the aircraft due to their inability to carry heavy equipment, contrary to what is mandatory in the current regulation. Excluding autonomous operations, the RP will be always responsible for applying the RoW rules while flying.

As this project perceives, for RPAS flying IFR in controlled airspace, the responsibility for the separation management will be in charge of ATC, and ultimately the RP for making decisions based on DAA recommendations. For RPAS flying in controlled or uncontrolled airspace under VFR, the RP will be entirely responsible for RWC from other aircraft supported by DAA systems, considering other resources such as traffic information services (if available) and always complying with current general flight rules.

1.4.1 EASA – EUROCONTROL: Low Level Flight Rules

In Europe, EUROCONTROL has partnered with EASA to develop the Low-Level Flight Rules (LFR) targeting U-Space operations [9]. These will be applicable in the VLL airspace, below the lowest VFR altitude (see [Fig 5](#)). It will include Visual Line of Sight (VLOS) and BVLOS operations. The LFR will regulate drones and RPAS in conjunction with VFR manned traffic, as they can also use this airspace. Intuitively, drones will have to be responsible for RWC of manned aviation, as the former are very difficult to 'see and avoid'. Similarly, BVLOS will need to have the RoW over VLOS, due to the complexity of the operation. LFRs have yet to be defined.

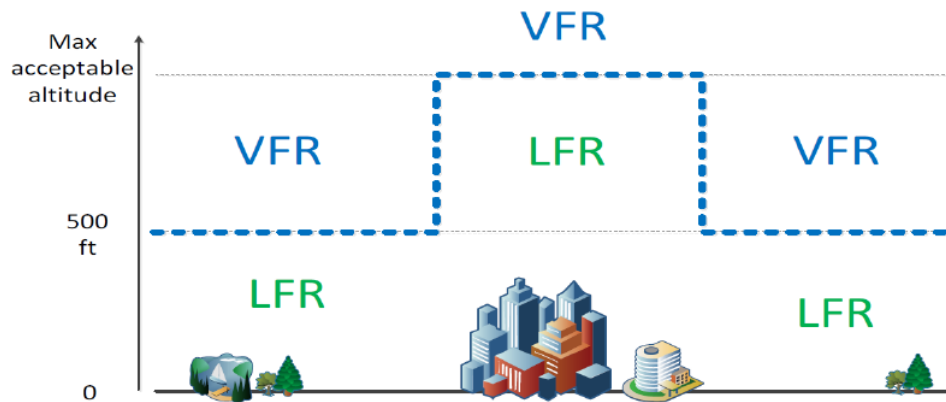


Fig 5 VFR/LFR Boundaries

CHAPTER 2. DETECT AND AVOID CONCEPT AND SPECIFICATIONS

Detecting and avoiding other traffic will become a key factor for the safe separation management of unmanned aircraft. In accordance to what has been explained in the previous chapter, current rules and systems do not adequately support aircraft operations that do not carry a pilot onboard. Enhancing the RP's situational awareness and suggesting safe dodging manoeuvres to avoid LoWC by means of more advance conspicuity devices could be the best solution. The ultimate goal is to provide RPs with the same capabilities as manned aircraft for any type of flight rules (IFR/VFR) and any type of airspace. DAA systems will enable RPAS and drones to RWC and avoid collisions with other traffic, frequently flying in a more complex environment. DAA is required to provide detection and guidance to maintain a well clear status and, when lost, to deliver recovery guidance to regain it.

In particular for the U-Space, DAA algorithms shall integrate solutions for the type of conflict geometries and situations occurring in the VLL. The fact that drones will typically conduct "mission-oriented" operations with unknown changes in their flight paths will create different conflict situations with traffic aircraft (i.e., very high vertical rates or more frequent multi-aircraft encounters). These operational differences require the development of adequate detection and alerting logics. Another important aspect in DAA systems for drones is the ability to avoid fixed obstacles, such as buildings, mountains or antennas. This ability should also be integrated into the manoeuvre guidance logic.

2.1 Remain Well Clear (RWC) Definition

The notion of Well Clear (WC) is directly linked to the International Civil Aviation Organization (ICAO)'s Rules of the Air and is stated as *"an aircraft shall not be operated in such proximity to other aircraft as to create a collision hazard"*. In contrast, RWC is *"the ability to detect, analyse and manoeuvre to avoid a potential conflict by applying adjustments to the current flight path in order to prevent the conflict from developing into a collision hazard"*, according to ICAO's Manual on RPAS [10].

It is important to highlight that WC and RWC are different concepts. WC is an aircraft state influencing the application of the right of way rules, whereas RWC should be understood as a separation minima between aircrafts, where its functions ensures that the aircrafts stay out of that minima.

There are currently no accepted time or distance-based standards for what it means for two aircraft to be RWC. That determination is left to the pilot's discretion.

2.2 Collision Avoidance (CA) Definition

Collision Avoidance in aviation is considered to be any manoeuvre made by the pilot or self-executed by electronic systems capable of deviating the trajectory of an aircraft to avoid a MAC, complying with the RoW rules. In DAA systems, if the RWC function does not mitigate the hazard, the CA function will override the RWC alerts and will issue CA alerts and associated manoeuvring guidance to ensure that the Near MAC (NMAC) Volume (NMACv) is not violated.

For RPAS flying in controlled airspace, the US Federal Aviation Administration (FAA) expressed the need to coordinate the manoeuvres proposed by ACAS systems (generally integrated into Mode S transponders) with DAA systems. According to the MOPS DO-365A [11], the coordination between both systems is established by alerting the TCAS II RA Directive alert (if the DAA system integrates TCAS II RA functionalities) [12]. In the VLL airspace though, there will be no aircraft carrying Mode S transponders due to its high cost and weight. Instead, the collision avoidance functionality will be integrated, coordinated and executed by the drone's autopilots.

Hereinafter, the definitions of DAA systems will be based on what expressed at European level through the ED-258 and also taking into account the work from USA through the MOPS DO-365A.

2.3 Mathematical Background and Metrics for DAA

Any separation assurance notion must include factors such as distance and time. These variables are functions over the current states of the aircraft that are compared to distance and time thresholds. The time of closest point of approach (t_{cpa}) and the distance at that time are the main time and distance variables in many conflict detection and resolution systems.

This subparagraph goes over some extra distance and time variables that are particularly relevant to the definition of a well-clear boundary model. In addition, there are the definitions and mathematical equations that support the detection and alerting criteria of a DAA system. Note that the core logic of new DAA systems is based on the TCAS II logic.

2.3.1 Time and Distance Metrics

The concept of "tau", which is calculated as the ratio of slant range between aircraft to their range rate measured in seconds, was first introduced in the Traffic alert and Collision Avoidance System (TCAS) collision detection logic to estimate the t_{cpa} between two aircraft. A vertical measure is also included in the TCAS detection logic that approximates the time until both aircraft are at co-altitude (t_{coa}). Vertical tau (τ_{ver}) is a metric that is determined by dividing the difference in altitude by the vertical range rate and is measured in seconds. But for encounters with a very low rate of closure, the calculated tau may be large while the physical

separation may be quite small. In these cases, a sudden change in one aircraft direction (i.e., a turn) could lead to a Loss of Well Clear (LoWC).

To overcome the previous issue, TCAS II uses a modified alerting threshold, often known as “modified tau” (τ_{mod}). This metric employs a new parameter called “Distance Modification” (DMOD) to establish a minimum range for alerting, independent of tau's estimated value. The same limit in the vertical domain is established by the “Vertical Separation Threshold” (ZTHR). These parameters define a cylinder -nicknamed ‘Hockey Puck’ by the MIT- around the aircraft that any intruder is allowed to enter (red cylinder in [Fig 6](#)).

Modified tau, on the other hand, has some drawbacks. For situations in which aircraft are on converging paths with a high rate of closure and a large miss distance, the modified tau metric will indicate an alert is required. TCAS II uses a “Horizontal Miss Distance” (HMD) to filter alerts for encounters where the separation at the Closest Point of Approach (CPA) exceeds the HMD parameter.

The “modified tau threshold” (TAUMOD) is the maximum horizontal time threshold of the well clear condition and is commonly measured as the sum of all the factors that delay the execution of a manoeuvre. The analogous threshold in the vertical domain is called “Time to Co-Altitude” (TCOA).

2.3.2 Hazard Alert Zone (HAZ) Definition

The Hazard Alert Zone (HAZ) is a hypothetical volume surrounding the aircraft that is used to know when to generate alerts in the event of potential conflicts between two or more aircraft. Such zone shall be considered as a reference volume from which the cascade alerts will be referenced. The hazard zone is based on a set of distance and time-based thresholds (TAUMOD, TCOA, DMOD, HMD and ZTHR) which defines the Well Clear Volume (WCV) - also known as Self-Separation Volume (SSV) or corrective volume. The ultimate goal of the DAA system is to prevent intruder aircraft entering the ownship's WCV.

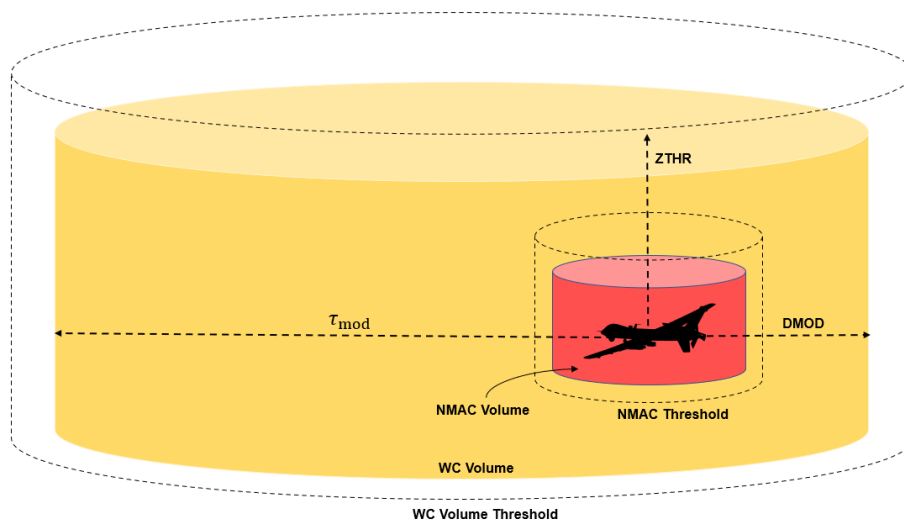


Fig 6 Well Clear Volume and Near Mid-air Collision Volume

Observe how the shape of the WCV is elongated in the direction of the aircraft speed vector (aircraft longitudinal axis) extending to the TAUMOD. It is purposely designed to ensure safe deconfliction manoeuvres at the highest relative speed, taking into account all possible manoeuvre delays (see [Fig 6](#)).

2.3.3 Hazard Zone Violation

If the appropriate distance and time variables determined by the relative aircraft state remain outside a set of predefined thresholds, then the two aircraft can be said to be "well clear" from each other. Three conditions must be met simultaneously to consider the situation as LoWC. The first condition is met when the horizontal separation between two aircraft during an encounter is less than the horizontal size of the HAZ, but is never less than DMOD (see [Fig 7](#)) [13]. This condition is analogous to be checking the τ_{mod} function; The second condition evaluates if the current path will result in an HMD, predicted by a constant velocity calculation; The third condition evaluates in the vertical domain whether the relative altitude is currently less than a threshold (ZTHR).

In general, there is a HAZ violation at Time i when:

$$[r \leq S^*] \text{ AND } [HMD_{p,i} \leq HMD^*] \text{ AND } [d_{h,i} \leq h^*] == TRUE \quad (2.1)$$

where:

- r is the current horizontal range between aircraft,
- S^* is the horizontal size of the HAZ for the alert type (the well clear τ_{mod} equation solved for range),
- $HMD_{p,i}$ is predicted Horizontal Miss Distance at CPA,
- HMD^* is the Horizontal Miss Distance threshold for the alert type,
- $d_{h,i}$ is the current Vertical Separation, and
- h^* is the Vertical Separation Threshold for the alert type (ZTHR).

The HAZ is violated for a given point in time when all three conditions are true. Horizontal Range (r) is defined as:

$$r = \sqrt{d_x^2 + d_y^2} \quad (2.2)$$

where:

- $d_x = x_2 - x_1$ is the current horizontal separation in the x dimension, and
- $d_y = y_2 - y_1$ is the current horizontal separation in the y dimension.

The horizontal size of the HAZ for a given alert type (S^*) is the value against which the horizontal range is compared, and is defined as:

$$S^* = \max \left(DMOD, \frac{1}{2} \left(\sqrt{(\dot{r} \tau_{mod}^*)^2 + 4DMOD^2} - \dot{r} \tau_{mod}^* \right) \right) \quad (2.3)$$

where:

$DMOD$ is the Distance Modification of Modified Tau,
 $\dot{r} = \frac{d_x v_{rx} + d_y v_{ry}}{r}$ is the horizontal range rate between the aircraft (negative for closing geometries),
 $v_{rx} = \dot{x}_2 - \dot{x}_1$ is the relative horizontal velocity in the x dimension,
 $v_{ry} = \dot{y}_2 - \dot{y}_1$ is the relative horizontal velocity in the y dimension, and
 τ_{mod}^* is the Modified Tau Threshold for the alert (TAUMOD).

In all cases:

$$\text{Horizontal Miss Distance (HMD}^*) = DMOD \tag{2.4}$$

Predicted Horizontal Miss Distance (HMD_p) is defined as:

$$HMD_{p,i} = \sqrt{(d_x + v_{rx} t_{CPA})^2 + (d_y + v_{ry} t_{CPA})^2} \tag{2.5}$$

where:

$t_{CPA} = \max(0, -\frac{d_x v_{rx} + d_y v_{ry}}{v_{rx}^2 + v_{ry}^2})$ is the time to Closest Point of Approach, positive for closing geometries.

The Horizontal Miss Distance Threshold (HMD^*) is the value against which the HMD_p is compared for a given alert type.

The Current Vertical Separation ($d_{h,i}$) is defined as:

$$d_{h,i} = \text{abs}(h_1 - h_2) \tag{2.6}$$

where h_1 and h_2 are the altitudes of the two aircraft. Vertical separation can be calculated using either reported barometric altitudes or true geometric altitudes.

The Vertical Separation Threshold (h^*) is the value against which the current vertical separation is compared for a given alert type.

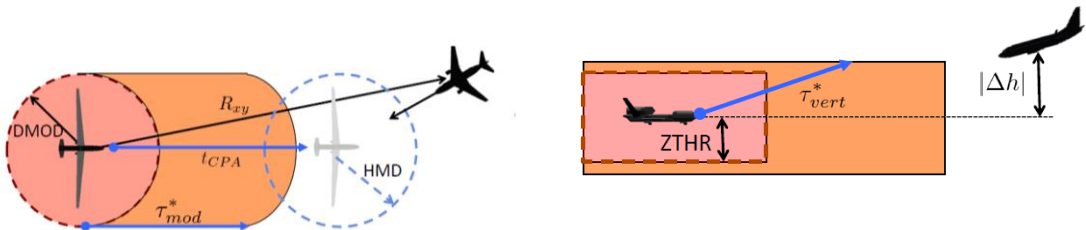


Fig 7 HAZ violation based on the Modified Tau criterion: Visual Description

2.4 Alert Types and Hierarchy

Alerts should come at optimal times to ensure that the RP is aware of the situation so that the WC status is not compromised. These alerts shall be followed by guidance proposals for safe manoeuvres in the case of potential conflicts, as defined by the RWC functions.

Alerts are sorted by level of priority and are executed according to the flight status of the ownship with respect to the intruder. Special care must be taken when defining the thresholds that activate these alerts to avoid a high level of nuisance. The alerts in [Table 2.1](#) are based on the MOPS DO-365A and are designed to raise situational awareness to pilots. In the case of completely autonomous aircraft operations, these alerts should be understood as time thresholds with which to feed the flight control algorithm.

Table 2.1 Alert Descriptions and Hierarchy

Alert	Description
Preventive	The preventive alert is the least restrictive, classified as an advisory alert. It is used to inform the pilot that the situation requires awareness and that if no action is taken, it could lead to a LoWC with an intruder aircraft.
Corrective	The corrective alert is triggered at the earliest point where the pilot is expected to begin manoeuvring to avoid entering the WCV (HAZ zone). It requests the pilot immediate awareness and possible corrective or compensatory actions. It is listed as a caution level alert.
Warning	This is the most restrictive alert. Once executed it warns the pilot that an immediate manoeuvre is required to prevent its aircraft enter the NMACv.

The preventive and corrective alerts shall be understood as part of the RWC function whilst the warning alert shall be considered as part of the CA function.

If the aircraft's DAA system is equipped with TCAS II RA functionality, then a fourth alert is aggregated, called TCAS II Directive RA. This alert is intended to inform the pilot that the autopilot is performing a coordinated and automatic manoeuvre as it has not been able to resolve the conflict by itself. The European Civil Aviation Equipment Organization (EUROCAE) has also considered this restrictive alert in its proposal for operational services for DAA systems in class D-G airspaces [14] (ED-258). They call it Auto CA (see [Fig 8](#)).

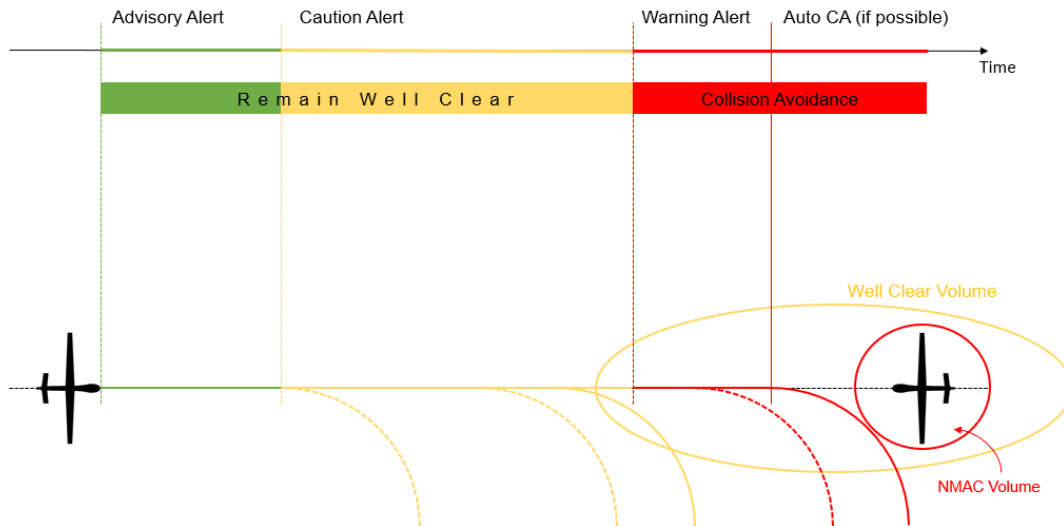


Fig 8 Alerting Process Timeline Concept

2.5 Manoeuvres to Remain Well Clear

The last fundamental part of a DAA system is the proposal of safe manoeuvres for the pilot to RWC. According to the MOPS DO-365A, a series of trajectory options in the form of manoeuvre guidance 'bands' are provided visually to the RP (see Fig 9). There are two types of bands. The generic bands which represent ranges of horizontal and vertical manoeuvres predicted to result in a HAZ violation. These are called negative guidance as following them will result in a LoWC. On the other hand, the recovery bands are a range of trajectories that must be followed (positive guidance) in order to regain WC status. In this case a HAZ violation is unavoidable. The development of manoeuvres is not part of this study.

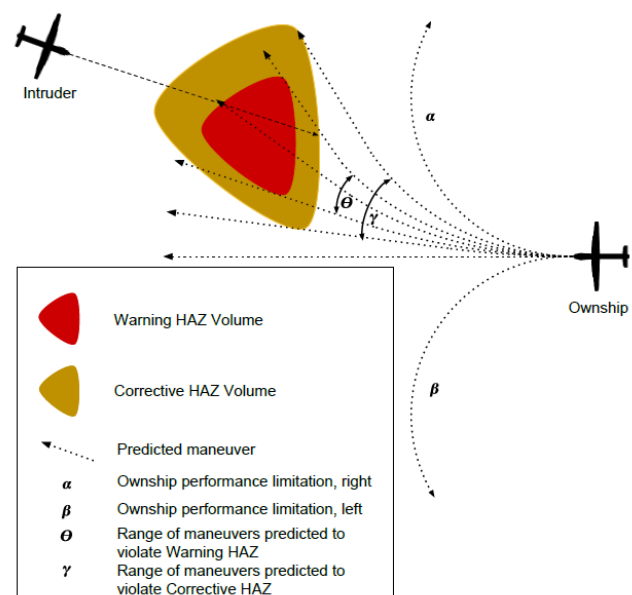


Fig 9 NASA's DAIDALUS Horizontal Manoeuvre Guidance Concept

2.6 DAA Equipment

Different DAA equipment is required depending on the airspace the aircraft is operating. For example, for RPAS flying in controlled airspace the onboard DAA system shall integrate current conspicuity devices such as Automatic Dependent Surveillance Broadcast (ADS-B) and/or Mode S transponders, for broadcasting and receiving from surrounding traffic positional information (detect) and avoiding collisions (avoid) by coordinating with current Collision Avoidance Systems (CAS) (TCAS II and future ACAS Xa). For RPAS and drones flying VFR receiving traffic information services (e.g., in class F airspace), DAA equipment could integrate FLARM technology or ADS-B Out for enhancing the RP's traffic awareness.

To date, still is unknown whether U-Space DAA systems will use ground systems or if this is solely a function of the aircraft depending on information it can receive from nearby traffic. In contrast, it is clear that drones and any other U-Space user must integrate electronic devices for scanning noncooperative intruders, that is, aircraft that do not transmit its location signal. Moreover, such systems shall also detect static obstacles and elevations in the terrain.

On the airborne section, the U-Space CAS will be limited by the range of the low Cost, Size, Weight, and Power (low C-SWaP) sensors and by the fact that they have to be in line of sight (LoS) with other intruders to be effective. Onboard sensors are typically non-cooperative in the sense that there is no communication between the two airborne systems. Some of them include Air-to-Air Radar (ATAR), Microwave Radars (MMW), Laser Identification Detection and Ranging (LIDAR), Electro-Optical (EO), Infrared (IR) or ultrasonic sensors, and thermal cameras (see [Fig 10](#)). Still is unclear what electronics will integrate cooperative systems and ground based surveillance facilities for the U-Space.

Future real-world implementations must ensure that the signal degradation of low C-SWaP sensors due to adverse weather conditions has no impact on airspace operability.



Fig 10 Example of LIDAR, electro-optical and ultrasonic sensors

CHAPTER 3. U-SPACE DETECT AND AVOID ALGORITHM

The next chapter details all of the changes that have enabled tuning the National Aeronautics and Space Administration (NASA)'s Detect and Avoid Alerting Logic for Unmanned Systems (DAIDALUS v.1.0) to the U-Space environment. DAIDALUS is a DAA software package, used as the reference implementation of DAA for unmanned aircraft systems in the MOPS DO-365. The objective is to provide a new set of detection volumes and alert thresholds adapted to typical U-Space situations and users, through tailored hazard volumes and alerts according to each aircraft's characteristics.

3.1 Aircraft Classification

Although the VLL airspace will be heavily dominated by sUAS, a few other aircraft will also be allowed to fly in that portion of airspace under justified conditions and carrying the necessary DAA equipment (i.e., light aircraft, Autonomous passenger Air Vehicles (AAV), emergency helicopters, police, armed forces, balloons, gliders, training and firefighting aircraft). Defining identical WCVs for all the users of this airspace would be inefficient due to the extensive difference in performance, nor would it be appropriate to adopt TCAS II DWC metrics as they are designed for bigger aircraft. Therefore, this project categorizes U-Space users by their flight dynamics and defines custom WCVs for each class (distances and time thresholds) in order to improve the airspace efficiency and capacity.

The data in the following subsections have been extracted from the manufacturer's technical datasheets of the most common sUAS/RPAS/light aircraft of each class. Four different classes have been identified based on the Rate of Turn (ROT) and the maximum speed, as these are considered the most important flight characteristics when studying RWC functionalities. A $1-\sigma$ filter has been applied to remove the outliers of all vehicles not complying with a one standard deviation from the normal distribution.

3.1.1 Class A

This class includes small and medium-sized tricopters, quadcopters, hexacopters and octocopters, commonly known as commercial drones. The most notable feature of this class is the high manoeuvrability, as they have the quickest attitude reaction among all aircraft. In fact, these drones are capable of stopping in the air and holding still, having an effective maximum ROT of 180 degrees per second. For simplicity, this manoeuvre has not been taken into account. The ROT versus maximum speed distribution is very uniform despite the differences in weight (see [Table 3.1](#)). The characteristic drone model of this class is the DJI MATRICE 600 PRO.

Table 3.1 Class A performance parameters.

CLASS A			
Brand & Model	MAX. RATE OF TURN (deg/sec)	MAX. SPEED (kts)	MTOM (kg)
DJI Inspire 2	150,00	50,76	4,00
TTA M4E	150,00	38,08	5,00
DJI Matrice 200 v2	150,00	43,74	6,14
DJI Matrice 300 RTK	150,00	44,71	9,00
DJI Matrice 600 pro	150,00	35,10	15,50
DJI Wind 4	150,00	34,99	21,00
DJI MG-1	130,00	42,76	24,00
WATTS Prism X8	120,00	40,14	25,00
DBG XYZ-802	120,00	36,11	35,00
YANGMAN 6 axis	100,00	29,64	45,00
μ	137,00	39,60	18,96
σ	18,29	6,08	13,73
$\mu + \sigma$	155,29	45,69	32,69
$\mu - \sigma$	118,71	33,52	5,24
Final Values	141,11	39,45	16,77

3.1.2 Class B1 & B2

These classes encompasses both fixed wing and Vertical Take-off Landing (VTOL) UAS, which can reach higher velocities but have less manoeuvrability. The Maximum Take-Off Mass (MTOM) has been used to split this class into two different classes (see [Table 3.2](#)). The characteristic UAS model of class B1 is the ALTI Ascend while for class B2 is the AEROSONDE from TEXTRON systems.

Table 3.2 Class B1 and B2 performance parameters.

CLASS B1			
Brand & Model	MAX. RATE OF TURN (deg/sec)	MAX. SPEED (kts)	MTOM (kg)
MyFlyDream MFD Nimbus 1800	40,00	48,60	5,00
Believer 1960mm	40,00	48,60	5,50
ALTI Ascend	50,00	48,60	9,00
Albatross UAV	50,00	77,75	10,00
Baby Shark VTOL	30,00	54,00	12,00
μ	42,00	55,51	8,30
σ	8,37	12,65	2,99
$\mu + \sigma$	50,37	68,16	11,29
$\mu - \sigma$	33,63	42,85	5,31
Final Values	45,00	49,95	8,17

CLASS B2			
Brand & Model	MAX. RATE OF TURN (deg/sec)	MAX. SPEED (kts)	MTOM (kg)
Aerosonde Fixed Wing	30,00	65,00	36,40
Insitu ScanEagle 3	40,00	80,09	36,30
Insitu Integrator	40,00	90,00	61,20
ALTI Reach	N/A	68,03	95,00
RQ-7B V2 Block III Shadow	N/A	107,99	212,00
μ	-	82,22	88,18
σ	-	16,89	77,73
$\mu + \sigma$	-	99,12	165,91
$\mu - \sigma$	-	65,33	10,45
Final Values	36,67	79,37	64,17

3.1.3 Class C

This last class includes manned aviation capable of flying within the VLL airspace (at or below 500ft AGL). These aircrafts have higher top speeds but less manoeuvrability compared to the above classes. For convenience, large military RPAS, gliders, hot air balloons or other related aircrafts are excluded from this class. Note that for this class, a standard ROT of 3 degrees per second is used (see [Table 3.3](#)). The characteristic aircraft of class C is the CESSNA 172.

Table 3.3 Class C performance parameters.

CLASS C			
Brand & Model	MAX. RATE OF TURN (deg/sec)	MAX. SPEED (kts)	MTOM (kg)
Robinson R22	-	102,00	622,00
Cessna T-41C	-	125,00	1134,00
Robinson R44	-	130,00	1134,00
Cessna 172	-	163,00	1157,00
Bell 206	-	120,00	1520,00
GippsAero GA8 Airvan	-	130,00	1814,00
Piper PA-46	-	213,00	1969,00
Eurocopter AS350	-	155,00	2250,00
Bell 407	-	140,00	2722,00
PAC P-750 XSTO	-	170,00	3402,00
μ	-	144,80	1772,40
σ	-	29,55	785,52
$\mu + \sigma$	-	174,35	2557,92
$\mu - \sigma$	-	115,25	986,88
Final Values	<i>Consider standard ROT of 3 deg/sec</i>	141,63	1568,29

3.1.4 Summary Table

This section summarizes the general performance parameters of the reference aircraft for each class.

Table 3.4 : General Performance Parameters by Class.

CLASS A	CLASS B1	CLASS B2	CLASS C
REFERENCE AIRCRAFT			
DJI MATRICE 600 PRO	ALTI ASCEND	AEROSONDE	CESSNA 172
MAX. BANK ANGLE (deg)			
78,90	64,10	69,61	21,27
RATE OF TURN (deg/sec)			
141	45	37	3
MAX. SPEED (kts)			
39,45	49,94	79,37	141,63
MTOM (kg)			
16,77	8,17	64,17	1568,29

The Bank Angle required to conduct a turn at a specific rate is directly proportional to True Airspeed (TAS) and Rate of Turn (ROT). The BA required to accomplish such maximum turns shown in [Table 3.4](#) has been computed as:

$$Bank\ Angle\ (deg) = \tan^{-1} \left(\frac{ROT \left(\frac{deg}{s} \right) \cdot V(kts)}{1091} \right) \quad (3.1)$$

where:

Rate of Turn (ROT) is in degrees per second, and
 V is the maximum True Airspeed (TAS) in knots.

3.2 RWC & CA Boundaries

Due to the absence of an extensive real drone trajectory database from which to extract information, this project is unable to simulate potential hazardous encounters using real data. This problem impairs the correct definition of the limits that define the WCV and NMACv. Thus, the definition of these limits in this project are based on previous studies carried out by competent organizations and a series of hypotheses based on the aircraft class, closure rate, performance characteristics and encounter geometry.

3.2.1 NMAC Volume (NMACv)

Historically, the NMAC volume for manned aviation has not changed much since its quantification in 1968. During TCAS development, the NMACv was quantitatively defined based on the size of the aircraft. Furthermore, it was imposed a 5:1 ratio between the horizontal and vertical distances without including any technical specification.

This project aligns with the historical NMAC definition for TCAS and the more recent MAC risk investigation conducted by Kochenderfer et.al. [15], both indicating that the probability of a NMAC depends on the physical size of the aircraft. The NMAC boundaries in this project (see [Table 3.5](#)) have been defined as follows:

For Class A, the horizontal and vertical distances are taken from the MIT Unmitigated Collision Risk Assessment for sUAS [16]. The result showed that the probability of an unmitigated MAC was around 10%, given the NMAC minima of 50 feet horizontally and 15 feet vertically. These distances comply with a recommendation given by Lester and Weinert stating that the specific quantitative means for a UAS to RWC of manned aircraft should be applicable if the sum of wingspan for an encounter between drones and General Aviation (GA) is 100 feet or less [17]. Considering that no light aircraft has a wingspan greater than 75 feet, the previous recommendation is met in this project.

The threshold values for classes B1 and B2 are assumed to be twice those specified for class A, as fixed-wing drones and VTOLs have less manoeuvrability.

Finally, the horizontal and vertical limits for class C are half the size of the values defined in the MOPS DO-365A. These values are higher to cope with pilots' reaction times and smooth manoeuvrings. For the sake of simplicity, all distance thresholds are multiple of 5 in feet.

Table 3.5 Proposed NMAC Volume parameters.

	Horizontal NMAC distance	Vertical NMAC distance
Class A	50 ft (15,24m)	15 ft (4,57m)
Class B1 & B2	100 ft (30,48m)	30 ft (9,14m)
Class C	250 ft (76,2m)	75 ft (22,86m)

3.2.2 Well Clear Volume (WCV)

The following section explains how the time and distance thresholds that make up the WCV have been obtained.

3.2.2.1 Time Metrics

TAUMOD and TCOA can be calculated as the sum of all the delays from when an intruder is detected by means of conspicuity devices until a dodging

manoeuvre is initiated by the ownship aircraft. This time span is made up of several pieces, including time for systems to process raw positioning data and time for the pilot to react (see [Table 3.6](#)). In a fully autonomous operation, the pilot's reaction time is assumed negligible.

Onboard systems are thought to scan for intruders at a frequency of 1 Hz, which is equal to one scan per second. It is estimated that the actions of detecting intruders, assessing the situation, suggesting alarms and safe manoeuvres lasts 2 seconds. The remote pilot's physical muscular reaction plus the decision to manoeuvre is estimated to take 4 seconds.

Table 3.6 DAA system processing time and pilot reaction time.

Action	Duration
Sensor Frequency	1 s
Track and Alert processing time	2 s
Pilot Reaction Time	4 s
Total RPAS (fully autonomous)	3 s
Total RPAS	7 s

Applying an extra margin of 3 seconds to add any other unforeseen factors we get TAUMOD (see [Table 3.7](#)). For convenience, this project avoids autonomous flights and asynchronous inputs of flight data, considering that all users report at the same time. Manned aircraft will be endowed with twice the TAUMOD than the other classes and TCOA will be considered negligible.

Table 3.7 Proposed TAUMOD and TCOA values.

	Preventive Alert			Corrective Alert			Warning Alert		
	Advisory			Caution			Warning		
	CLASS A	CLASS B1&B2	CLASS C	CLASS A	CLASS B1&B2	CLASS C	CLASS A	CLASS B1&B2	CLASS C
TAUMOD (seconds)	10	10	20	10	10	20	10	10	20
TCOA (seconds)	0	0	0	0	0	0	0	0	0

3.2.2.2 Distance Metrics

The CORUS consortium has already proposed a set of minimum WCV limits for drones. These distances are given by BVLOS interaction between two drones, without taking into account manned aviation.

NASA have also quantified these boundaries based on military RPAS trajectories provided by the NASA UAS Track Database. NASA's default values are referred to the standard use-case described in the MOPS for DAA, focused on the integration of especially large military RPAS in segregated airspace, but in any case, considering typical speeds, sizes and heights of the VLL airspace.

The Massachusetts Institute of Technology – Lincoln Labs (MIT LL) from its part, has proposed a series of well clear boundaries exclusively based on distances. Their studies address sUAS operations up to 1200ft with MTOM up to 25kg, including conflicts with LpGA.

Table 3.8 CORUS, DO-365 and MIT WCV thresholds.

	CORUS	DO-365A (from NASA)	MIT LL
DMOD	200ft (60,96m)	4000ft (1219,2 m)	2000ft (609,6m)
HMD	200ft (60,96m)	4000ft (1219,2 m)	2000ft (609,6m)
ZTHR	150ft (45,72m)	450ft (137,16 m)	50ft (76,2 m)

Note that the MOPS DO-365A values in [Table 3.8](#) are excessive since they consider greater speeds, pilot reaction times, interaction with ATC and manoeuvring times.

Until now, the minimum distances that ensure aircraft to maintain safe separation with surrounding traffic are independent of the type of aircraft involved in the conflict. As stated in the MOPS DO-365A, a single well clear volume, alert and guidance processing scheme ensures integrity for all aircraft. This condition is no longer applicable within U-Space, due to the large differences in performance between U-Space users. The boundaries that make up the hazard alert zone should vary according to the type of intruder aircraft. Consequently, each aircraft should integrate dynamic well clear volumes, updated on-the-fly in real time depending on an intruder type classification.

This project adopts the intruder-centric model suggested by NASA [18], where each intruder aircraft is assigned a WCV and alerting scheme depending on the size of the vehicle. The largest aircraft of the encounter will be considered as intruder. For example, if a Class A drone approaches a Class C helicopter, then the intruder aircraft (helicopter) will be assigned a Class C scheme (see [Table 3.9](#)).

For demonstrating that the WCV dimensions should vary depending on the intruder aircraft type, this project has simulated different encounter scenarios with very different conditions and aircraft combinations using a dedicated software tool (Refer to [APPENDIX A. DYNAMIC WELL CLEAR VOLUME DEMONSTRATION](#))

The distance thresholds suggested in this project are the following: Class A metrics corresponds to BVLOS interaction between two drones, adopted from the CORUS research; Class B1 and B2 metrics are slightly enlarged to cope with greater speeds and lower manoeuvrability; Class C metrics are taken from the MIT LL investigation, which has considered conflicts between drones and LpGA.

Table 3.9 Proposed DMOD and ZTHR parameters

Alert Type	Alert Level	Class Type	DMOD = HMD*	h* = ZTHR
Preventive	Advisory	CLASS A	200ft (60,96 m)	150ft (45,72 m)
		CLASS B1	300ft (91,44 m)	150ft (45,72 m)
		CLASS B2	400ft (121,92 m)	150ft (45,72 m)
		CLASS C	2000ft (609,6 m)	450ft (137,16 m)
Corrective	Caution	CLASS A	200ft (60,96 m)	100ft (30,48 m)
		CLASS B1	300ft (91,44 m)	100ft (30,48 m)
		CLASS B2	400ft (121,92 m)	100ft (30,48 m)
		CLASS C	2000ft (609,6 m)	300ft (91,44 m)
Warning	Warning	CLASS A	200ft (60,96 m)	100ft (30,48 m)
		CLASS B1	300ft (91,44 m)	100ft (30,48 m)
		CLASS B2	400ft (121,92 m)	100ft (30,48 m)
		CLASS C	2000ft (609,6 m)	300ft (91,44 m)

3.3 Tracking and Alerting Process

The DAA algorithm used in this project is a modification of NASA's DAIDALUS software, adapted to the U-Space environment. The rationale is based on TCAS II Resolution Advisory (RA) logic. DAIDALUS includes algorithms for determining the current well clear status between two aircraft, predicting a LoWC within a lookahead time, computing the time interval of well clear violation and generating alerts given the severity of the WC violation. [APPENDIX B. COMPREHENSIVE DESCRIPTION OF DAIDALUS ALGORITHM](#) contains a comprehensive description of the DAA algorithm used in this project.

3.3.1 Well Clear Logic

The well clear logic tries to specify if the intruder aircraft is violating the hazard zone of the ownship aircraft at the initial moment of the collision course. To accomplish so, it examines if the proper distance and time variables indicated by the relative aircraft state remain outside of a set of established thresholds, using the mathematical functions shown in Section [2.3.3](#).

3.3.2 Detection Logic

Using a predefined lookahead time, the detection logic predicts a time interval for each hazard volume violation between the ownship and the intruder aircraft. Because these projections are calculated at constant speed (linear), the fewer

manoeuvres the ownship aircraft perform, the more accurate the prediction will be.

In essence, the intruder's trajectory is projected linearly, and the intersection of that trajectory with the parametric line of the ownship's hazard zone is determined. This parabola-shaped perimeter can be defined using a quadratic equation, the determinant and roots of which will determine whether the area intersects or not and the entry and exit timings, respectively.

3.3.3 Alerting Logic

The alerting logic assigns one alert to each aircraft indicating its current alert level based on a set of time and distance boundaries provided as input configuration parameters. First, the entry and exit times found during the detection process are compared against a set of predefined alert times (see Section [3.3.3.1](#)). A preventive, corrective or warning alert is triggered depending on these intervals. The hierarchical alerting schema can be found in Section [2.4](#).

3.3.3.1 Alert Times

With respect to the alert type, early, late, and minimum average time alerts are defined. These time intervals establish a time limit within which the RP will be alerted of the possibility of a forecast LoWC (or, eventually, a MAC) and advised to conduct a dodging manoeuvre.

Several Human-In-The-Loop (HITL) and Real Time Simulations (RTS), largely conducted by NASA, suggested the alert times throughout the development of the MOPS DO-365A [19]. The provided numbers in this project (see [Table 3.10](#)) are estimates based on pilot attitudes, aircraft performances and flight dynamics since no HITL or RTS were able to be executed. These alerts are in compliance with the TAUMOD criterion (see Section [3.2.2](#)).

Table 3.10 Proposed Alert Time Thresholds.

Alert Type	Preventive			Corrective			Warning		
Alert Level	Advisory			Caution			Warning		
Class Type	CLASS A	CLASS B1 & B2	CLASS C	CLASS A	CLASS B1 & B2	CLASS C	CLASS A	CLASS B1 & B2	CLASS C
Minimum Average Time of Alert (seconds)	20	20	30	20	20	30	12	12	22
Late Threshold (seconds)	10	10	20	10	10	20	5	5	15

Early Threshold (seconds)	30	30	40	30	30	40	20	20	40
--	----	----	----	----	----	----	----	----	----

CHAPTER 4. MULTI-ROTOR DRONE ENCOUNTER MODEL

When designing and evaluating new DAA systems, it is vital to test their effectiveness with close encounters that could lead to a collision. Because there is a lack of real conflicts involving any type of drone, the encounter modelling technique allows us to generate a large number of artificial drone trajectories which may end up in a MAC. These artificial encounters are then used in FTSS to predict how new safety nets will perform in simulated operational scenarios. In that sense, this project has developed the Drone Encounter Generator (DEG) tool.

DEG is used to generate random pairwise (two drones) trajectories between non-cooperative medium and small size drones in the final stages before a collision. The way in which trajectories are formed is strongly influenced by the type of operation that is performed. In DEG there are two types of operation, inspection or surveillance flights and logistics flights. Since the aircraft are noncooperative (do not carry a transponder) and no ATC is considered prior to the conflict, any change in one aircraft's trajectory has no effect on the behaviour of other aircraft. Hence, DEG is an uncorrelated encounter model.

The encounter variables of this model are an estimation of the real conditions of future U-Space environments, and are stochastically sampled –Monte Carlo sampling- from a probabilistic distribution table. Alternatively, the encounter situations of this model are abstracted in the sense that there is no consideration of an explicit location or local airspace structure. Refer to [APPENDIX D. DEG ARCHITECTURAL DESIGN](#) to find DEG's structural design.

4.1 Architectural Design

There are four processes involved in creating encounters with DEG (see [Fig 11](#)). First, DEG defines the desired trajectory by generating segments. It starts from the CPA (or NMAC point) and then generates segments forwards and backwards. These segments specify the initial and target (or final) altitudes and speeds, as well as the flight duration for each interval of the trajectory, based on high-level flight actions: three horizontal manoeuvres (cruise, turn, and hover) and three vertical manoeuvres (level, climb and descent). Afterwards, DEG calculates the 4D waypoints that will make up the intended trajectory by making linear projections using the precalculated parameters. DEG then uses a dedicated Python package called PyDy to identify the quadrotor's kinematics and dynamics, allowing it to mimic the drone model's flight through each 4D waypoint. During the simulation, the quadrotor's state derivatives (position, speed, and heading) are updated from waypoint to waypoint by controlling the drone's attitude with a PX4 autopilot. Lastly, DEG rotates the intruder's path to match with the required CPA encounter parameters.

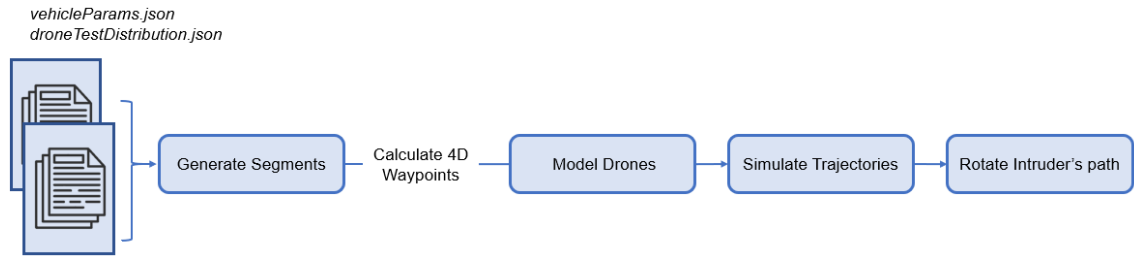


Fig 11 DEG Encounter Generation Procedure

4.1.1 Generation of Segments

The generation of segments starts by Monte Carlo sampling (random stochastic sampling) the initial encounter conditions from a probabilistic distribution table. These variables are Vertical Miss Distance (VMD), Horizontal Miss Distance (HMD), approach angle, altitude and speed, which are going to be used to generate the NMAC segment containing the CPA point. Then, the initial and target (or final) altitude and speed as well as the flight duration are determined based on certain high-level flight actions (cruise, climb, descend, turn or hover). The steps for generating the NMAC segment are hierarchically listed in [Table 4.1](#).

Table 4.1 Steps for Generating the NMAC Segment

Initialize CPA conditions	Set initial conditions at CPA by Monte Carlo sampling NMAC values (VMD, HMD, AppAngle, Altitude (Own,Intr), Speed (Own,Intr), ClassIdx (Own, Intr), Class (Own,Intr)).
Generate NMAC Segment	<p>Determine NMAC segment parameters:</p> <ul style="list-style-type: none"> - Get a unitary form value (NMACFactor) ranging from 0 to 1. This will specify NMAC location as a percentage of total NMAC segment. - Determine the horizontal mode (1: constant, 2: turn, 3: hover). - Compute segment duration and speed variation (1: constant, 2: accelerate, 3: decelerate). If speed mode is constant or horizontal mode is hover then the segment duration is sampled from the distribution. - Get initial and target (final) speeds of NMAC segment using same percentage provided by NMACFactor. If the target speed is greater than 0kts and the next segment horizontal mode is hovering, then it is calculated the time and distance required to stop the drone having a deceleration factor of G/2. - Assign heading variations if in a turn. The heading variation and the rate of turn are sampled from the distribution tables. - Determine vertical mode (1: level, 2: climb, 3: descent). If no level, computes altitude variation using NMACFactor. - Compute the target altitude, altitude variation and updates the segment duration depending on the vertical mode and the vehicle's performance. - Add segment to segment list.

Next, the transition distribution variables are Monte Carlo sampled. The forward and backward segments are generated following the same logic, but taking into consideration the initial and final parameters of the adjacent segments. Note that the dynamics at this stage are simply obtained by discrete sampling from the probability distributions and do not take into consideration the actual 6-DOF quadcopter dynamic model specified in [Section 4.1.3.1](#). The steps for generating the forward and backward segments are hierarchically listed in [Table 4.2](#).

Table 4.2 Steps for Generating Forwards and Backwards Segments

Generate Forward Segments	<p>Determine forward segment parameters:</p> <ul style="list-style-type: none"> - Determine the horizontal mode (1: constant, 2: turn, 3: hover). - Compute segment duration and speed variation (1: constant, 2: accelerate, 3: decelerate). If previous segment has null horizontal speed, then the next segment speed mode will be to accelerate. If speed mode is constant or horizontal mode is hover then the segment duration is sampled from the distribution. - Calculate speed variation and target speed for each segment according to vehicle performances and speed modes. If the target speed is greater than 0kts and the next segment horizontal mode is hovering, then it is calculated the time and distance required to stop the drone having a deceleration factor of $G/2$. - Assign heading variations if in a turn. The heading variation and the rate of turn are sampled from the distribution tables. - Determine the vertical mode (1: level, 2: climb, 3: descent). If level, altitude variation is null. - Compute the target altitude, altitude variation and updates the segment duration depending on the vertical mode and the vehicle's performance. - Add segment to segment list. <p>The initial altitude, speed and heading of forward segments correspond to the final (target) parameters of the previous segment.</p>
Generate Backward Segments	<p>Backward segments are determined using the same logic as for forward segments but these are inserted as first elements of the list. The final (target) altitude, speed and heading of backward segments correspond to the initial parameters of the next segment on the list.</p>
Add initial Segment	<p>Creates the initial segment. The drone starts hovering then accelerates to reach the initial conditions of the first segment of the list (the last backward segment). The total duration of the initial segment is 5 seconds.</p>

Location of the Closest Point of Approach

For convenience, the NMAC segment will start at the Cartesian Coordinate (0,0,0) m. The exact location of the CPA point (or NMAC point) within the NMAC segment is defined as a percentage of the total NMAC segment using a unit value called the NMAC factor. Note that the intruder's altitude at CPA is the ownship's altitude plus VMD.

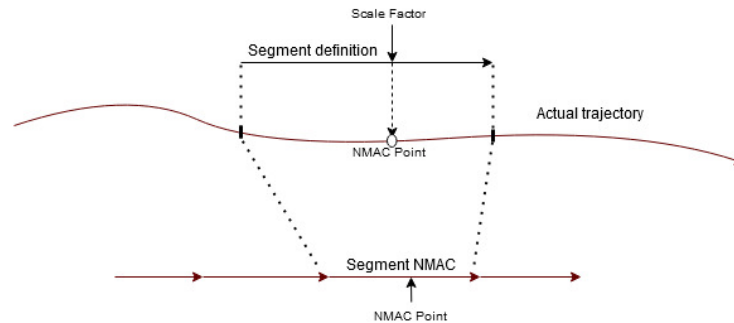


Fig 12 CPA (or NMAC) Segment Generation Concept

Probability Distributions

DEG creates segments based on the likelihood that specific high-level flying activities could be performed (straight segments, climb/descent, hold, enroute turns, mission scanning turns, etc.). The way in which segments are formed is heavily influenced by the type of operation being flown. The precise generation of each of the segments is determined by the actual vehicle performance as well as the limits of the sampled vehicle class.

There are two types of operations in DEG:

1. Inspection or surveillance flights with substantial heading and vertical changes that can be seen in common area analysis patterns.
2. Logistic flights, including both human and cargo air transportation, with minor vertical or course adjustments.

The magnitude and probability of occurrence associated with each flight action, as well as the conditions at the CPA, reflect a realistic prediction of the future U-Space environment. These variables are stochastically sampled from a discrete probability distribution table. This solution may be the most popular because of its realism and its capacity to generate as many encounters as needed.

4.1.2 4D Waypoints

Once the starting and ending speeds, altitudes and heading changes as well as the time duration for each segment are determined, then the desired 4D waypoints are calculated. A 4D trajectory refers to a sequence of waypoints that consist of desired 3D cartesian coordinates and reliable and reachable timestamps at which a flight reaches these points. [Table 4.3](#) contains the detailed explanation of the 4D waypoint calculation procedure.

Table 4.3 4D Waypoints Calculation

<p>Calculate NMAC 4D Waypoints</p>	<p>Calculates the initial and final positions of the CPA segment expressed in 4D cartesian coordinates.</p> <ul style="list-style-type: none"> - If straight segment, a 2D velocity vector is calculated based on the target speed and initial heading parameters. This vector along with the segment duration allows for the computation of the 3D spatial coordinates using a simple linear projection. - If turning, the segments are divided into subsegments of 45 degrees of heading change (see Fig 13). Finally, it is calculated the total segment duration and target positions based on the subsegments. - If hovering, the initial and target altitudes and segment durations are obtained from the initial NMAC segment parameters. <p>Fig 13 120 Degree heading change in steps of 45 degrees.</p>
<p>Calculate Forward and Backward 4D Waypoints</p>	<p>The same logic is applied for calculating 4D waypoints forwards and backwards. For forward segments, the starting 3D coordinates and time of the new segment correspond to the ending position and time of the previous segment; For backward segments, the target 3D coordinates and time of the new segment correspond to the starting position and time of the next segment.</p>

4.1.3 Drone Modelling

There are two quadcopters available in DEG, namely DJI F450 (see Fig 14) and DJI F450 FAST. The characteristics of the two drones are configured to match the real DJI F450, which is a small, low-power commercial entertainment drone. The performance envelope of each model is shown in Table 4.4.



Fig 14 DJI F450 Quadcopter.

Table 4.4 Drone Limits and Operational Envelope.

	Aircraft Model	
	DJI F450	DJI F450 FAST
MTOM (kg)	1,2	1,8
Dimensions (cm x cm)	16x16	16x16
Max. Speed (m/s)	4,8	11,1
Min. Speed (m/s)	0	
Max. Acceleration (m/s ²)	1,14	5,28
Max. Deceleration (m/s ²)	-4	-4,44
Max. Climb Rate (m/s)	4,8 *	11,1
Max. Descent Rate (m/s)	4,8 *	11,1
Motor thrust coeff. (N/(rad/s) ²)	1.076e-5	
Motor torque coeff. (Nm/(rad/s) ²)	1.632e-7	
Min. Thrust (N)	0,4	4
Max. Thrust (N)	36,7	367,2
Min. Rotor Rotation Speed (rad/s)	75	150
Max. Rotor Rotation Speed (rad/s)	925	2000

*If path angle is greater than 42.5 degrees when climbing; or greater than 30 degrees when descending

The fast version has more than 50% better performance in terms of top speed and acceleration. Its PID controller has been tuned to cope with its faster attitude responses. The main differences between both drones are shown in [Table 4.5](#).

Table 4.5 Differences in Performance Characteristics Between Models.

		Aircraft Model	
		DJI F450	DJI F450 FAST
Motor Limits	Min. Thrust (N)	0,4	4
	Max. Thrust (N)	36,7	367,2
	Min. Rotor Rotation Speed (rad/s)	75	150
	Max. Rotor Rotation Speed (rad/s)	925	2000
	Max. Certified Speed Envelope (Never Exceed Speed) (m/s)	5	12
PID Controller	Position Gain (Pz)	4	2
	Horizontal Speed Gain (Px dot)	14	15
	Horizontal Speed Gain (Dx dot)	1,4	2
	Horizontal Speed Gain (Ix dot)	14	15
	Vertical Speed Gain (Pz dot)	12	15
	Vertical Speed Gain (Dz dot)	0,4	3
	Vertical Speed Gain (Iz dot)	14	10
	Rate Proportional Gain (Pp)	1,5	0,5
Rate Derivative Gain (Dp)	0,04	0,01	
Speed	Max. Horizontal Speed (m/s)	4,8	11,1
	Max. Lateral Speed (m/s)	4,8	11,1
	Max. Vertical Speed (m/s)	4,8	11,1
	Max. Velocity All (m/s)	4,8	11,1

4.1.3.1 Drone Model Kinematics and Dynamics

The aircraft model uses a 6 Degree of Freedom (6-DOF) system with 18 state variables, 12 for position, rotation, linear and angular speeds and 8 for motor speeds. This 6-DOF system is the most complex yet most accurate aircraft model since allows the representation of the three translational and three rotational movements of the rigid body (see [Fig 15](#)). This project utilizes PyDy and SymPy libraries for building the complex multibody aircraft system, and for applying Kane's method to derive the symbolic equations of motion.

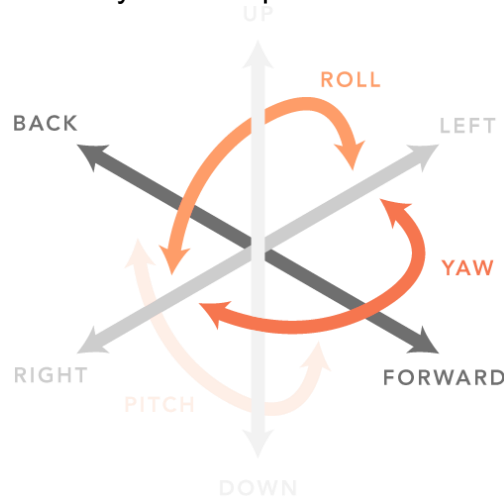


Fig 15 Translational and Rotational Movements of a 6-DOF Model

Geometric Assumptions

DEG software is able to express the drone's orientation either in the North-East-Down (NED) frame or East-North-Up (ENU) frame. For convenience, this project utilizes the NED frame. Here, the inertial reference system is oriented in such a way that the X direction is North, Y is East and Z is Down. The drone's orientation in this frame is front-right-down. NED frame is extensively used in aeronautics, and also the frame used for the PX4 multicopter controller.

The rotation of the drone is expressed using quaternions, because it eliminates the sine and cosine functions in the equations of motion. As the equation of motion has to be computed many times during a single timestep of the simulation, the use of quaternions to describe the drone's rotation significantly decreases the computing time.

State Variables

There are twelve state variables that completely define the aircraft dynamic model. Position, rotation (Euler angles), linear and angular velocities are established by three variables each while the rotation in quaternions is defined using four variables (see [Table 4.6](#)).

Table 4.6 : Aircraft Dynamic State Variables

TYPE	PARAMETER	UNITS
Position	x	Meters [m]
	y	
	z	
Rotation (Euler Angles)	ϕ	Degrees [°]
	θ	
	ψ	
Rotation (Quaternion)	qw	[-]
	qx	
	qy	
	qz	
Linear Velocity	x $\dot{}$	Meters/Second [m/s]
	y $\dot{}$	
	z $\dot{}$	
Angular Velocity	p	Radians/Second [rad/s]
	q	
	r	

Motor Dynamics

In order to simulate the motor's dynamics (2nd Order System), eight additional state variables are added to the quadcopter model. Four variables for motor angular velocities and four for accelerations (see [Table 4.7](#)).

Table 4.7 Motor Dynamics State Variables

TYPE	PARAMETER	UNITS
Motor Angular Velocity	wM1	Radians/Second [rad/s]
	wM2	
	wM3	
	wM4	
Motor Angular Acceleration	w $\dot{}$ M1	Radians/Square Second [rad/s ²]
	w $\dot{}$ M2	
	w $\dot{}$ M3	
	w $\dot{}$ M4	

4.1.4 Trajectory Simulation

The previously calculated 4D waypoints set the desired trajectory for each drone. A simple interpolation using a time scaling factor provides the intermediate sequence of positions, making the final trajectory. A PID control system integrating a PX4 control algorithm, allows the model to navigate through each waypoint. For each time step the quadrotor dynamic model states are updated, the desired states are computed and the control commands for the next iteration are calculated. The refresh rate is set to 50Hz. The trajectory's total duration can vary. Even though the total duration is limited to 90 seconds (60 seconds before the CPA and 30 seconds after the CPA), the last segments continue until the desired manoeuvre is finished.

Control

The control algorithm is a replica of the PX4 multi-copter control algorithm. It is a cascade controller, where the position error (difference between the desired position and the current position) generates a velocity setpoint, the velocity error then creates a desired thrust magnitude and orientation, which is then interpreted as a desired rotation (expressed as a quaternion). The quaternion error then generates angular rate setpoints, which then creates desired moments. The states are controlled using a PID control. Position and Attitude control uses a simple Proportional (P) gain, while Velocity and Rate uses Proportional and Derivative (D) gains.

a. PID Controller

A PID controller is a robust electronic control mechanism which maintains the output of a system such that there is zero error between the processed variable and the desired output by closed-loop operations. PID uses three basic control behaviours: Proportional (P), Integral (I), and Derivative (D). The proportional value depends on the current error, the integral depends on the past errors, and the derivative is a prediction of future errors. In the case of a drone, these three actions together allow the regulation of the angular velocity of the electric motors to control the positioning of the drone.

An extensive analysis has been carried out to tune the PID controller gains so that the drone has the fastest and most robust response and the lowest possible damping factor, always being in the region of stability. Refer to [APPENDIX C](#) to see the analysis in detail.

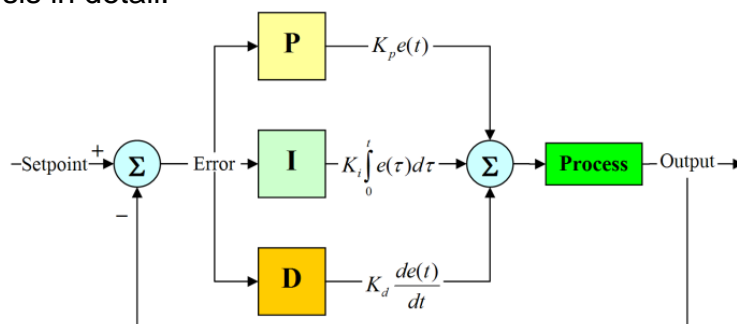


Fig 16 PID Close-loop Feedback Illustration

4.1.5 Intruder's path rotation

DEG simulates the two trajectories independently of the CPA conditions that are stochastically sampled at the beginning. To fulfil the CPA criteria once the ownship and intruder's trajectories are generated, DEG adjusts the intruder's trajectory geometry and duration. This technique is carried out in two stages: first, the two aircraft's speeds, locations, and angles are determined at CPA. Then, these variables are translated, rotated, and timely adjusted for the intruder trajectory, to fit with the required CPA circumstances.

CHAPTER 5. RESULTS

This chapter details the results found during the completion of this project. First, an example trajectory generated with DEG is presented. Then, an evaluation of the performance and effectiveness of the DEG tool will be conducted.

5.1 DEG: Conflict Trajectory Example

This section shows a conflict trajectory example generated with DEG. The conflict involves two small DJI F450 quadcopter drones. The intruder is a delivery drone, while the ownship drone performs area inspection missions. The colour codes of the graphics are the following:

- / ● 4D Waypoints
- CPA point
- Desired trajectory (connecting 4D waypoints)
- Actual flight path
- NMAC Segment (containing the CPA point)
- Ownship's flight path
- Intruder's flight path

CPA (or NMAC) Conditions

The CPA parameters stochastically sampled from the distribution tables are summarized in [Table 5.1](#).

Table 5.1 DEG Generated Encounter: Initial CPA Conditions

VMD	12,336 m	(40,472 ft)
HMD	6,303 m	(20,679 ft)
Ownship altitude	37,470 m	(122,933 ft)
Intruder altitude	49,806 m	(163,406 ft)
Ownship speed	0 m/s	(0 kts)
Intruder speed	1,557 m/s	(3,027 kts)
Approach angle	-156,496 deg	

Ownship Aircraft: Desired Trajectory & Actual Flight Path

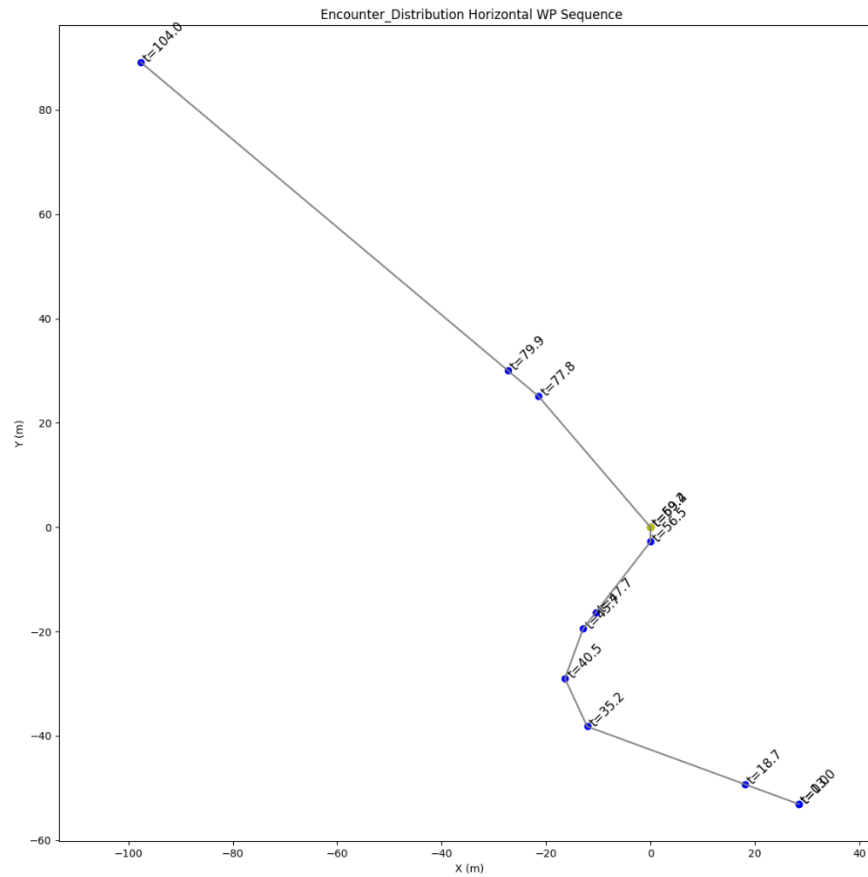


Fig 17 Ownship Desired Trajectory: Horizontal View

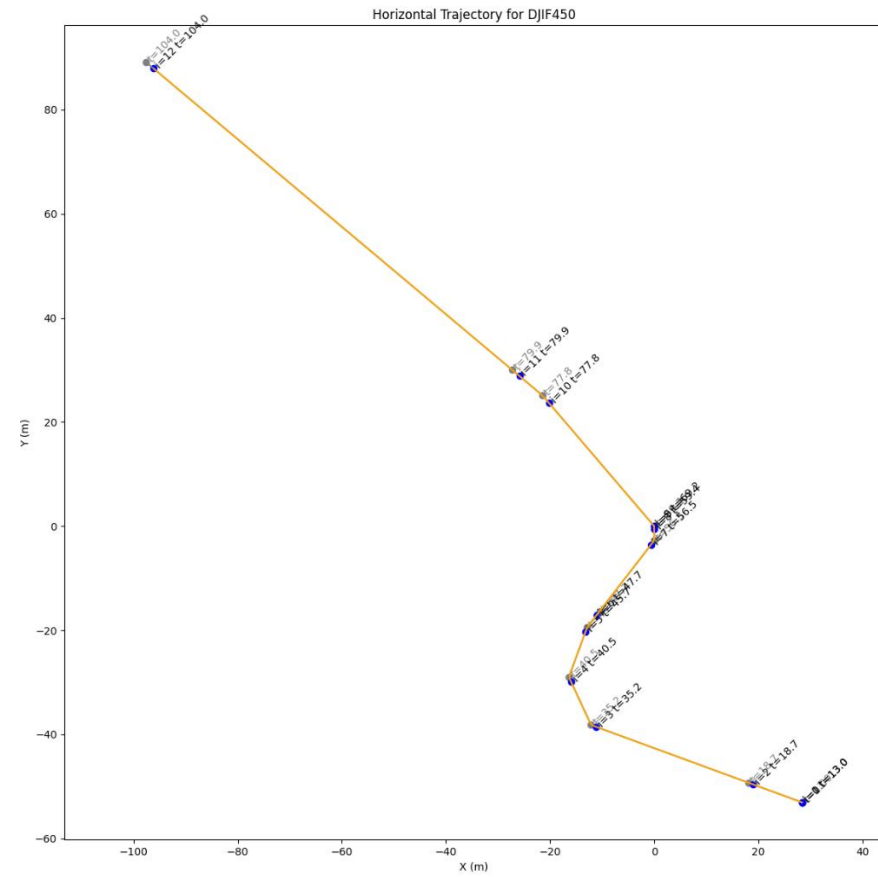


Fig 18 Ownship Actual Flight Path: Horizontal View

Observe how the drone flies more distance than desired in the actual flight path (orange line in Fig 18). The arrival times for each waypoint are clearly ahead of schedule. This is due to very fast PID responses in the autopilot (see Fig 20).

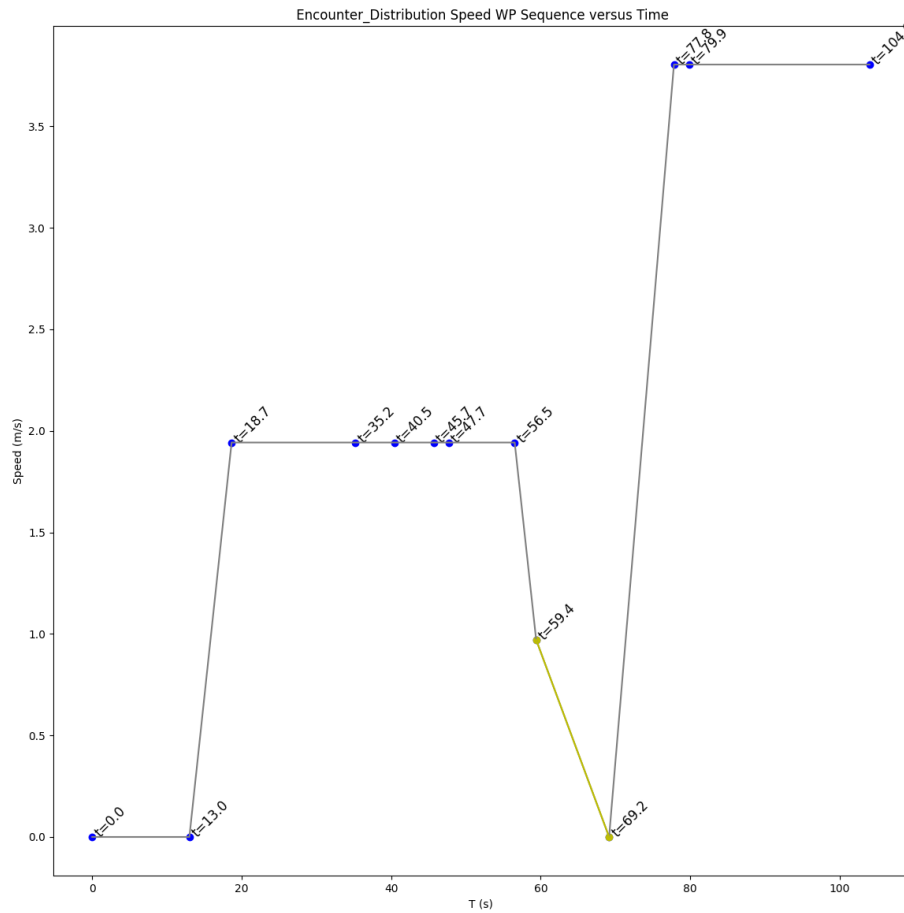


Fig 19 Ownship Desired Trajectory: Speed vs Time

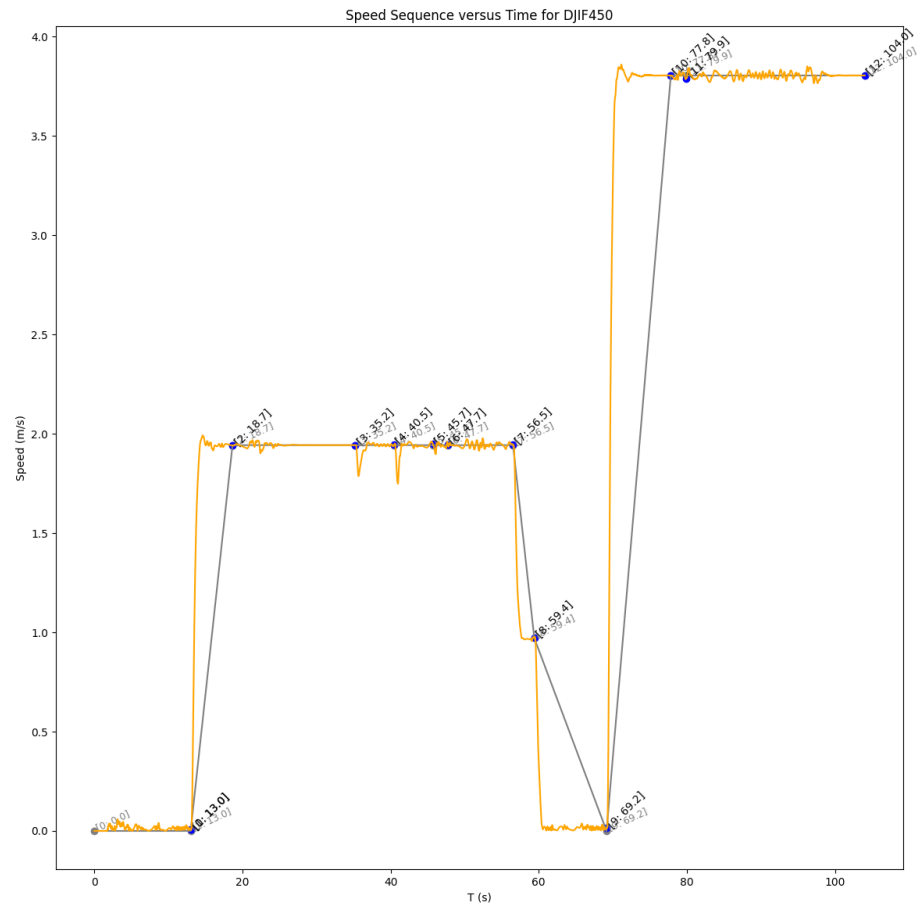


Fig 20 Ownship Actual Flight Path: Speed vs Time

The autopilot is not able to accelerate/decelerate continuously (the PID controller has a very fast response). Hence, the drone reaches the desired speed too early, flying further than expected. Note that the system is stable (underdamped).

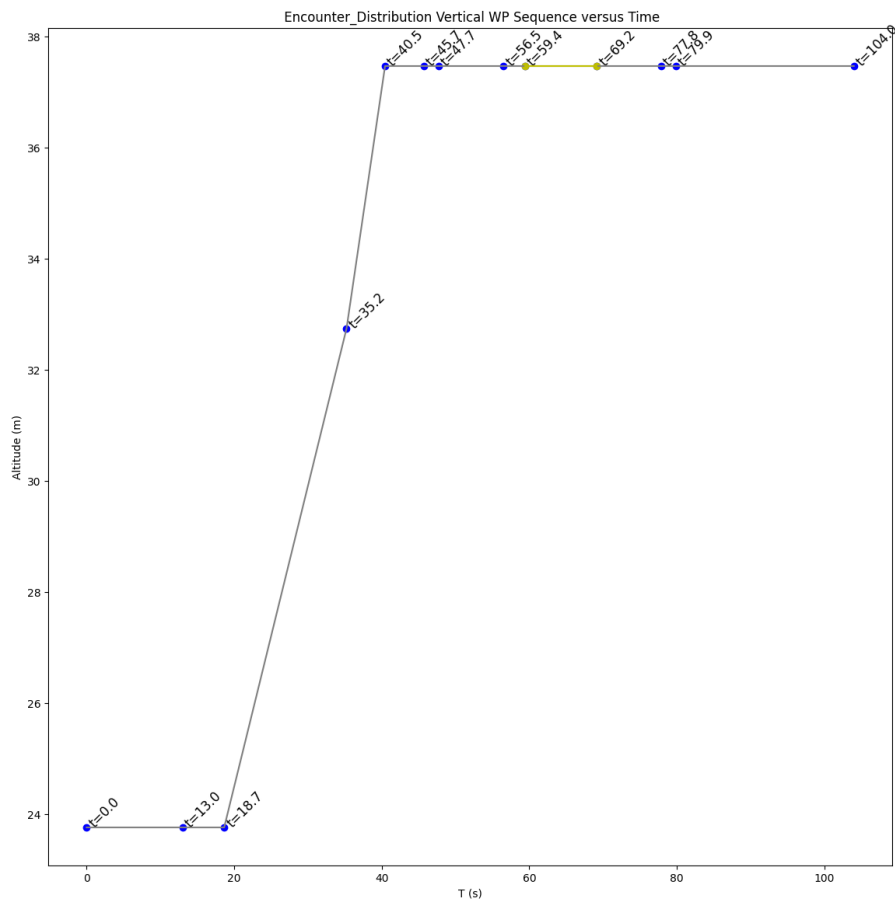


Fig 21 Ownship Desired Trajectory: Altitude vs Time

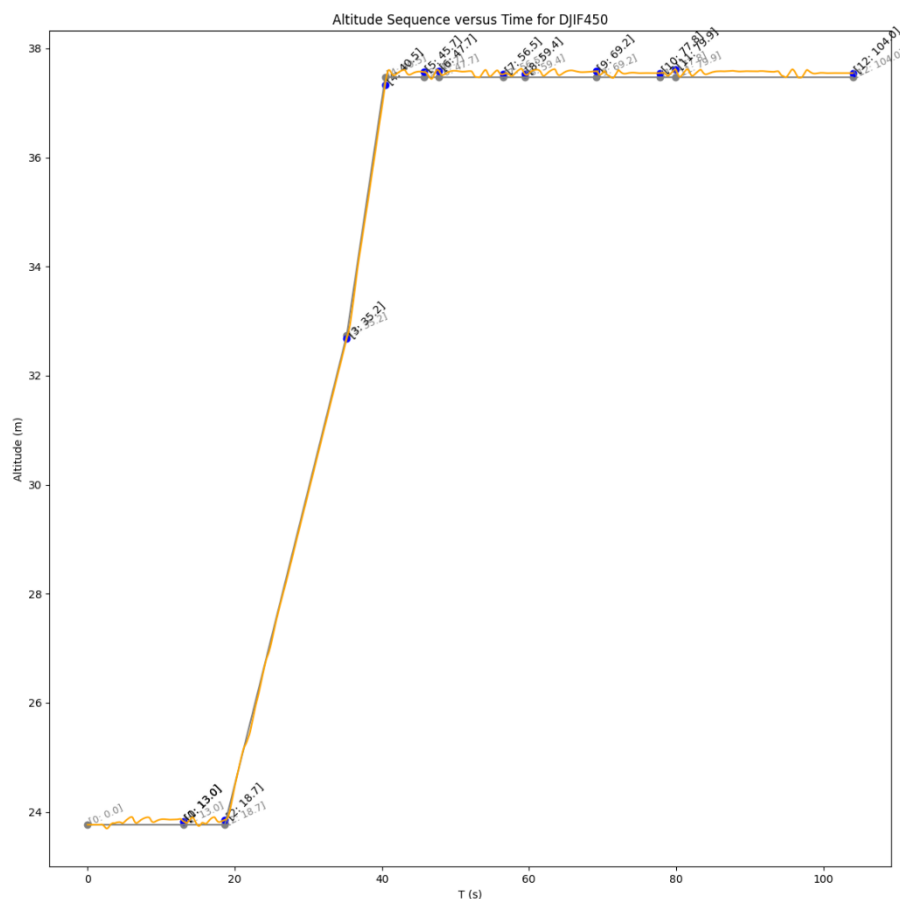


Fig 22 Ownship Actual Flight Path: Altitude vs Time

Minor altitude variations can be observed in the actual flight path due to speed oscillations when transitioning between waypoints.

Intruder Aircraft: Desired Trajectory & Simulated Flight

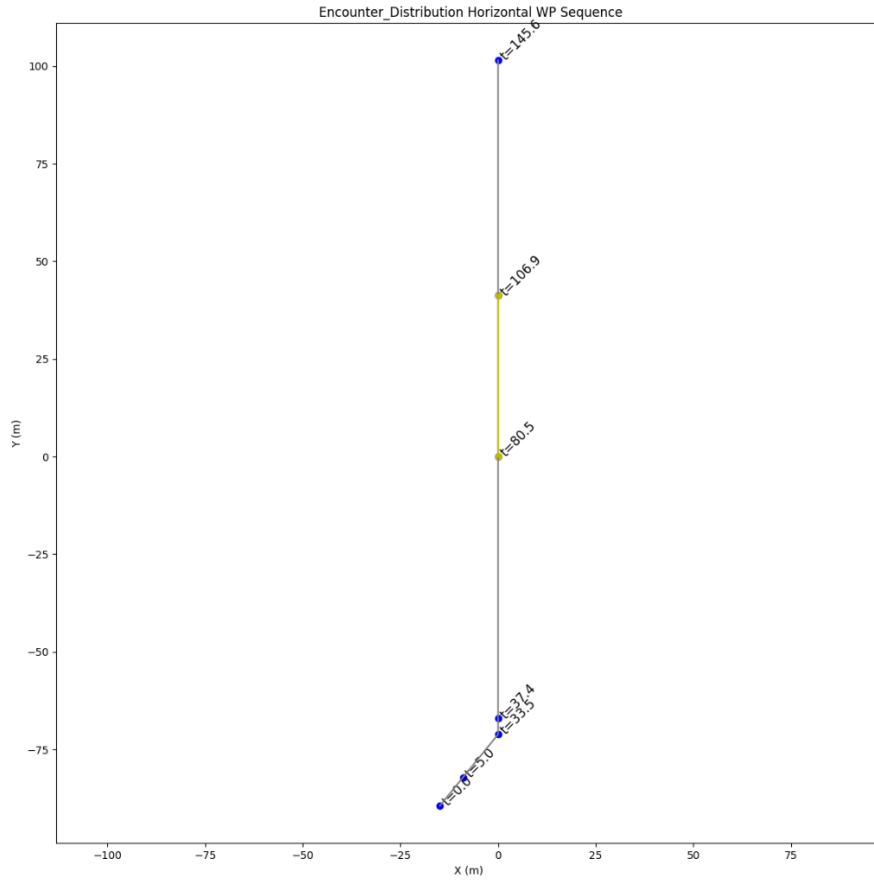


Fig 23 Intruder Desired Trajectory: Horizontal View

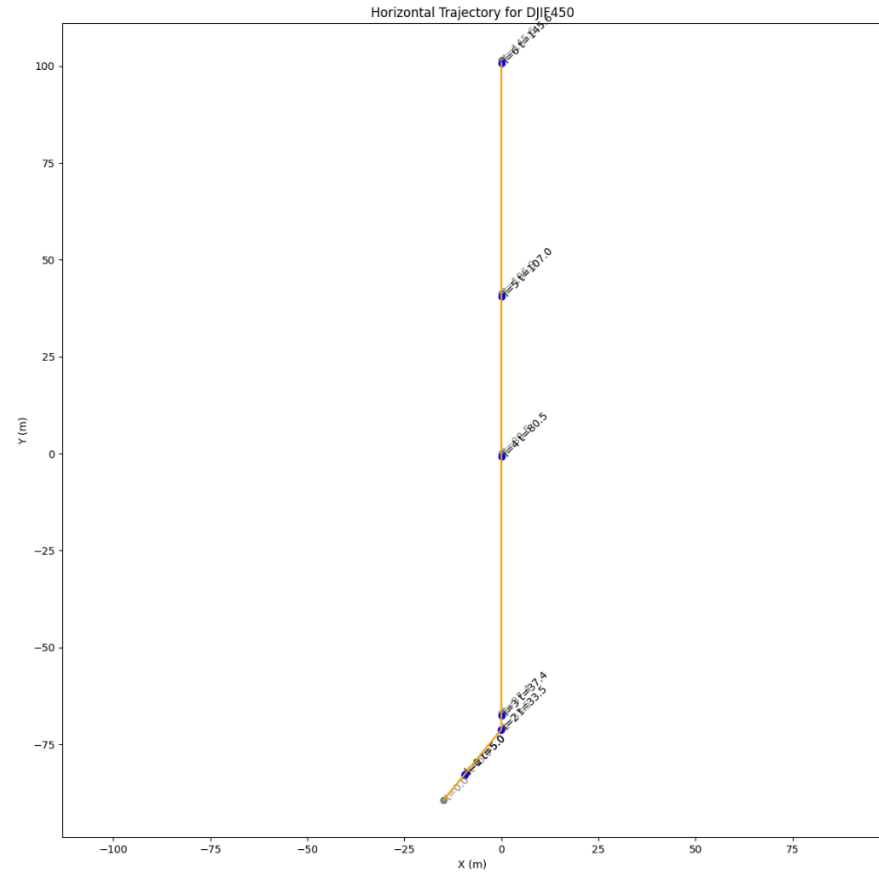


Fig 24 Intruder Actual Flight Path: Horizontal View

The intruder's actual flight path (orange line in Fig 24) is identical to the desired trajectory.

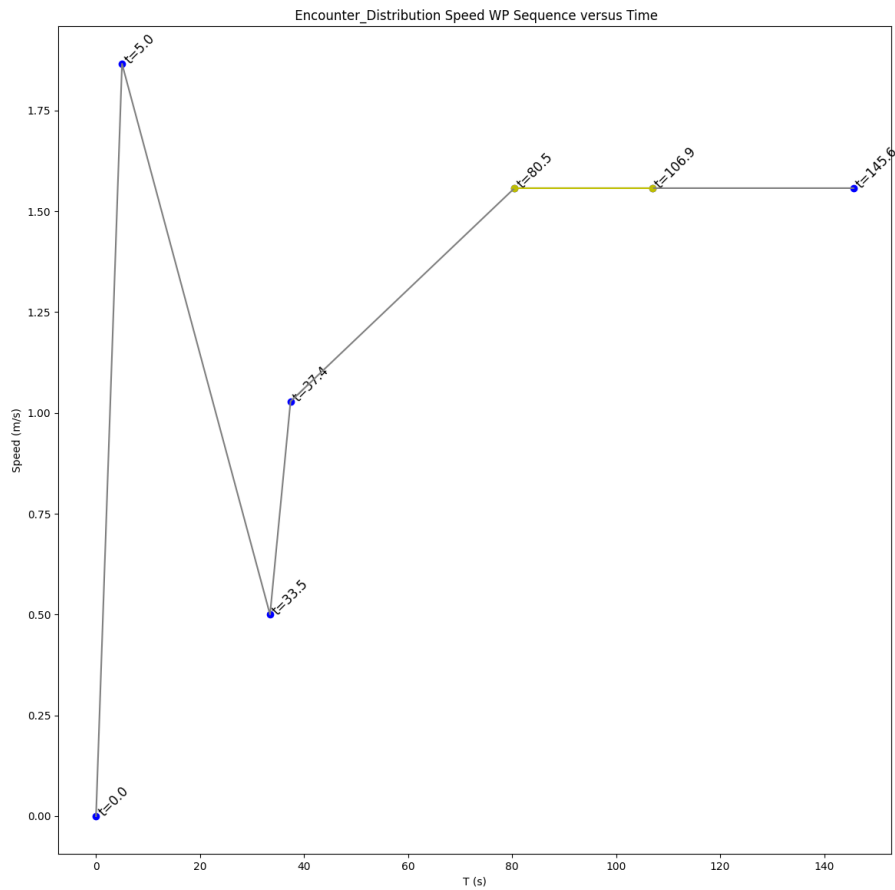


Fig 25 Intruder Desired Trajectory: Speed vs Time

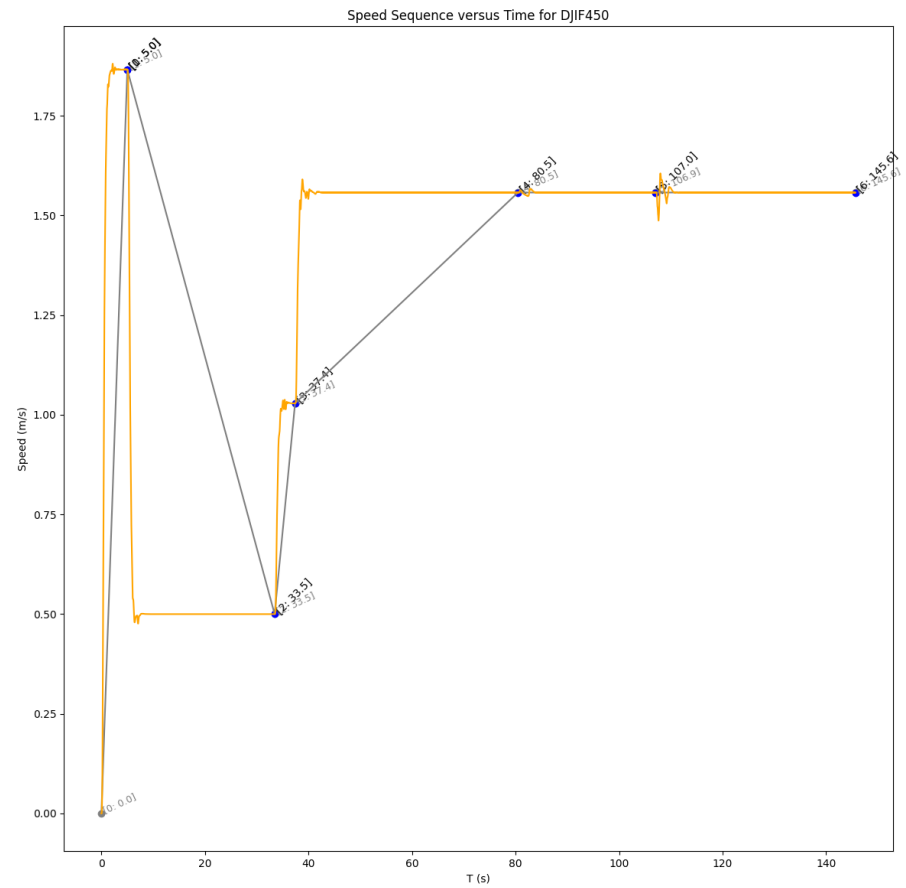


Fig 26 Intruder Actual Flight Path: Speed vs Time

As previously explained, the PID controller commands do not allow the drone to accelerate/decelerate at a continuous rate. Nevertheless, the actual flight path is still accurate (see Fig 24).

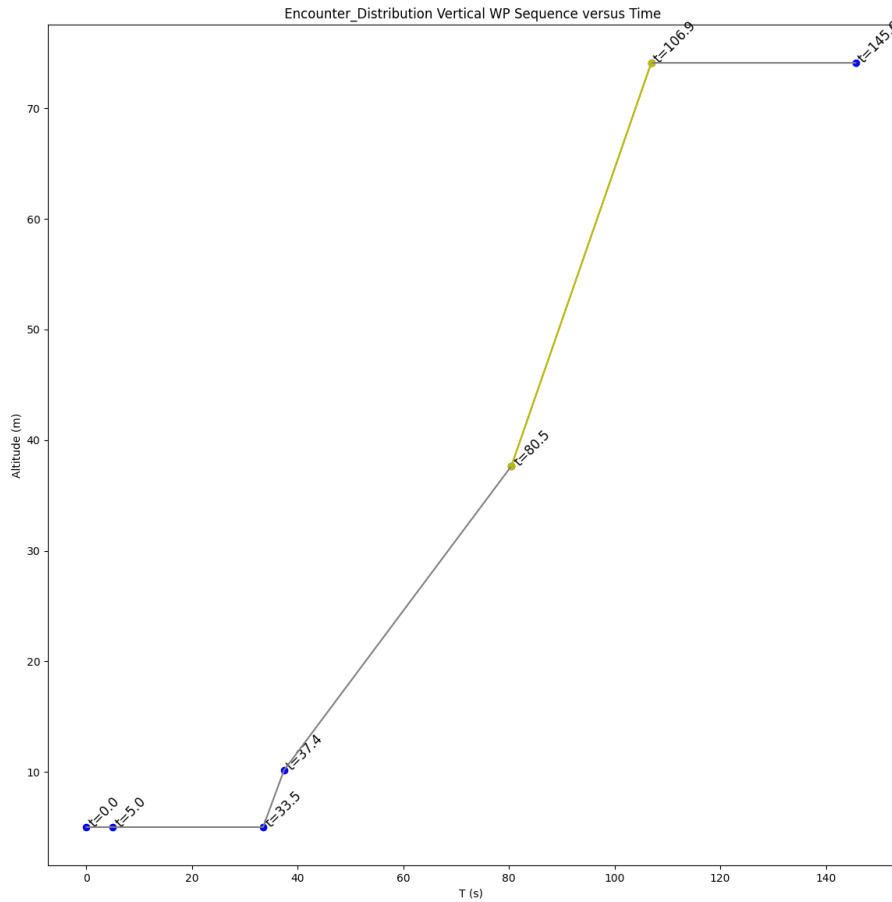


Fig 27 Intruder Desired Trajectory: Altitude vs Time

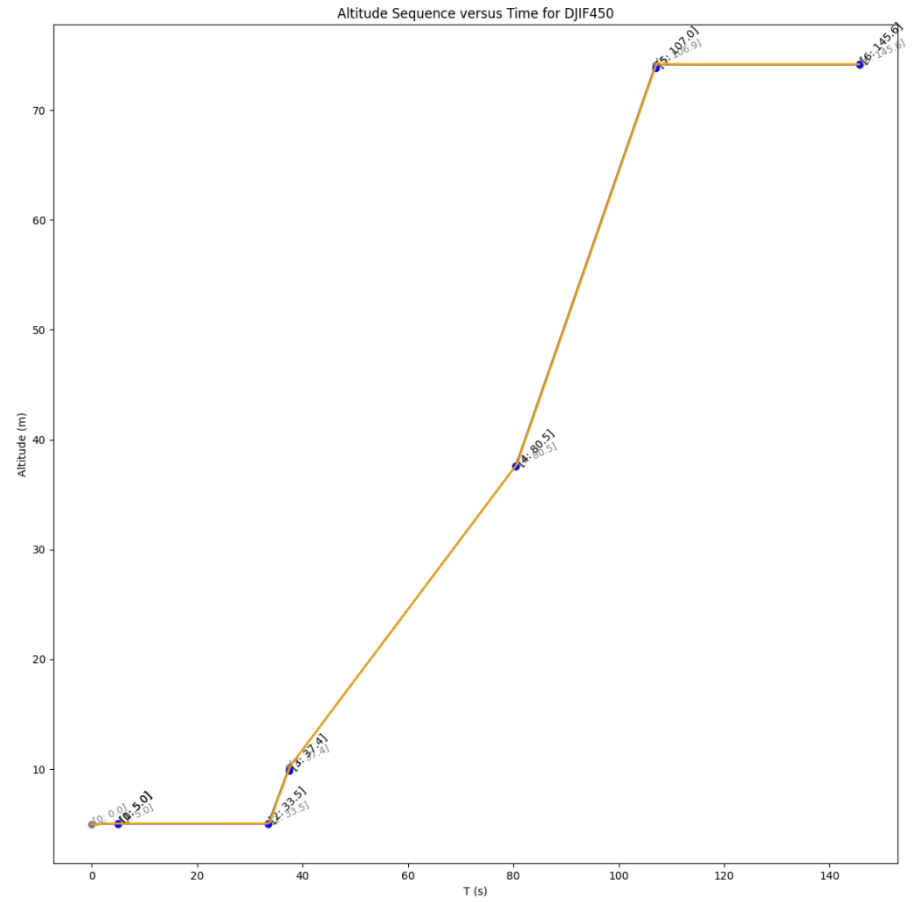


Fig 28 Intruder Actual Flight Path: Altitude vs Time

Note that there are no altitude oscillations in the actual flight path since the transition between waypoints is very smooth in this case.

DEG Generated Encounter: Ownship (Green) vs Intruder (Blue)

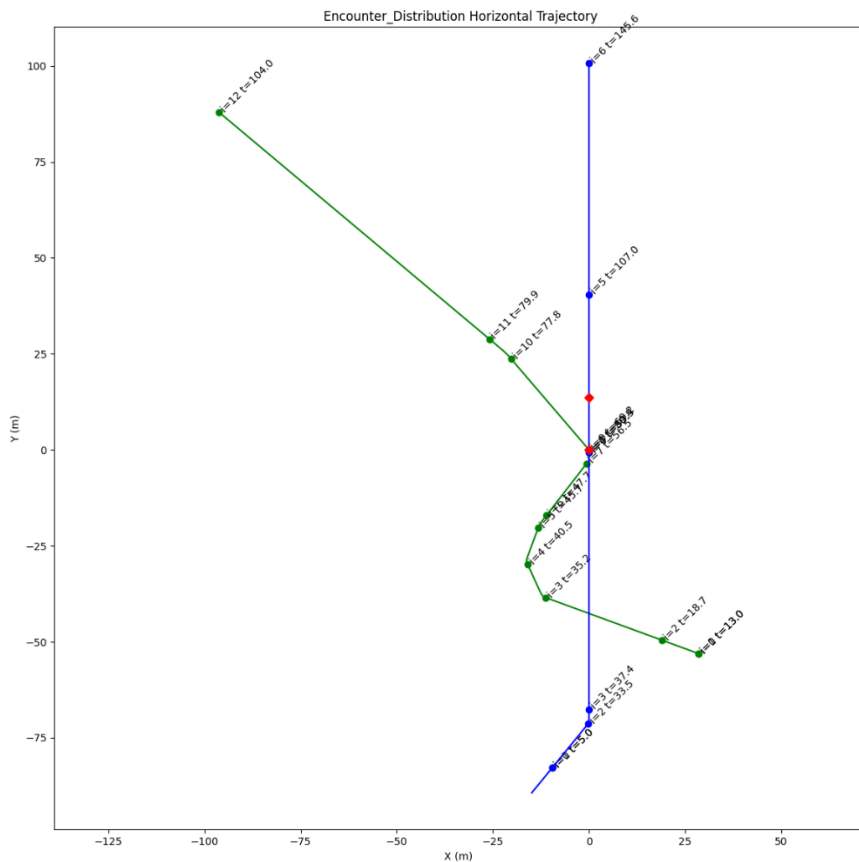


Fig 29 DEG Encounter: Horizontal View

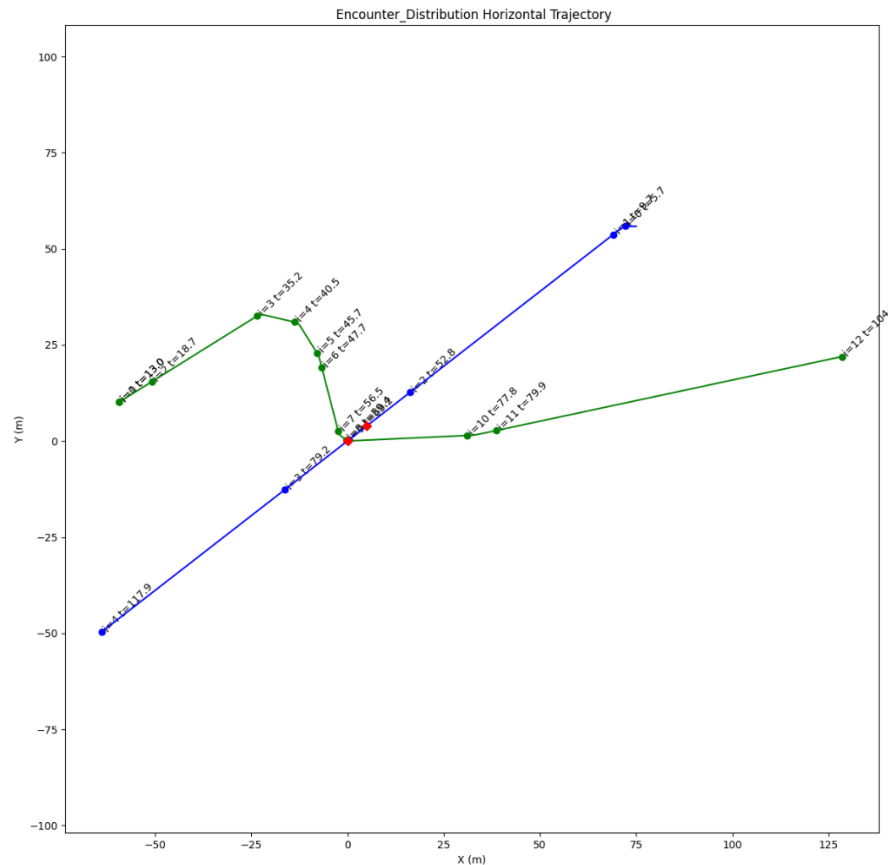


Fig 30 DEG Rotated Encounter: Horizontal View

The rotated encounter (Fig 30) is considered as the final trajectory. We can see that the sampled CPA conditions (Table 5.1) are accomplished (HMD = 6,303 m and Approach Angle = -156,496 deg).

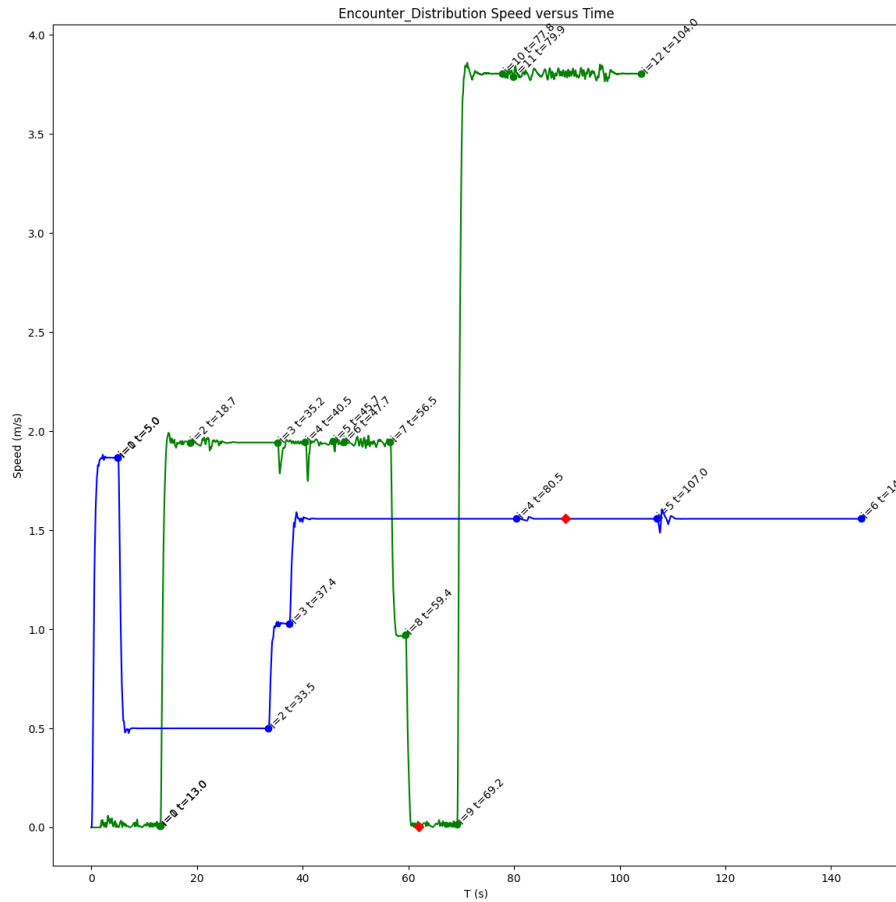


Fig 31 DEG Encounter: Speed vs Time

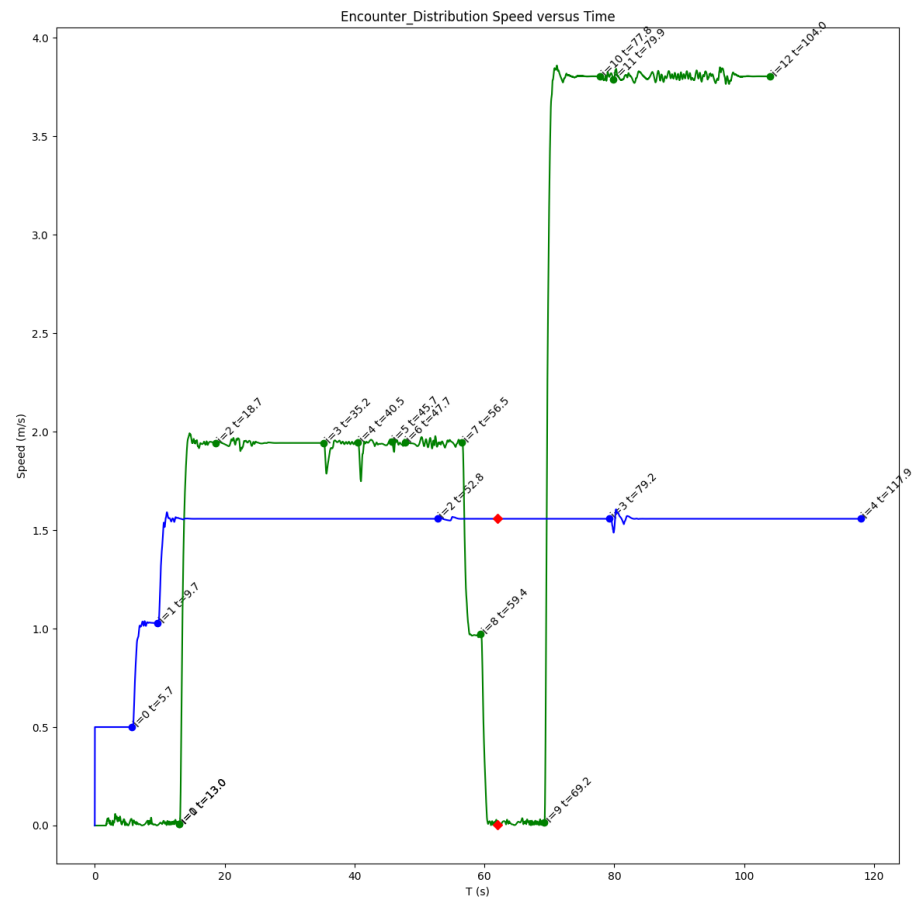


Fig 32 DEG Rotated Encounter: Speed vs Time

Observe in the rotated encounter (Fig 32) how the CPA point (red dots) is aligned in time for both aircraft. This readjustment in time modifies the intruder's speed distribution (blue line in Fig 32). Again, the sampled CPA conditions are accomplished (Ownship speed (green line) = 0 m/s; Intruder speed (blue line) = 1,557 m/s).

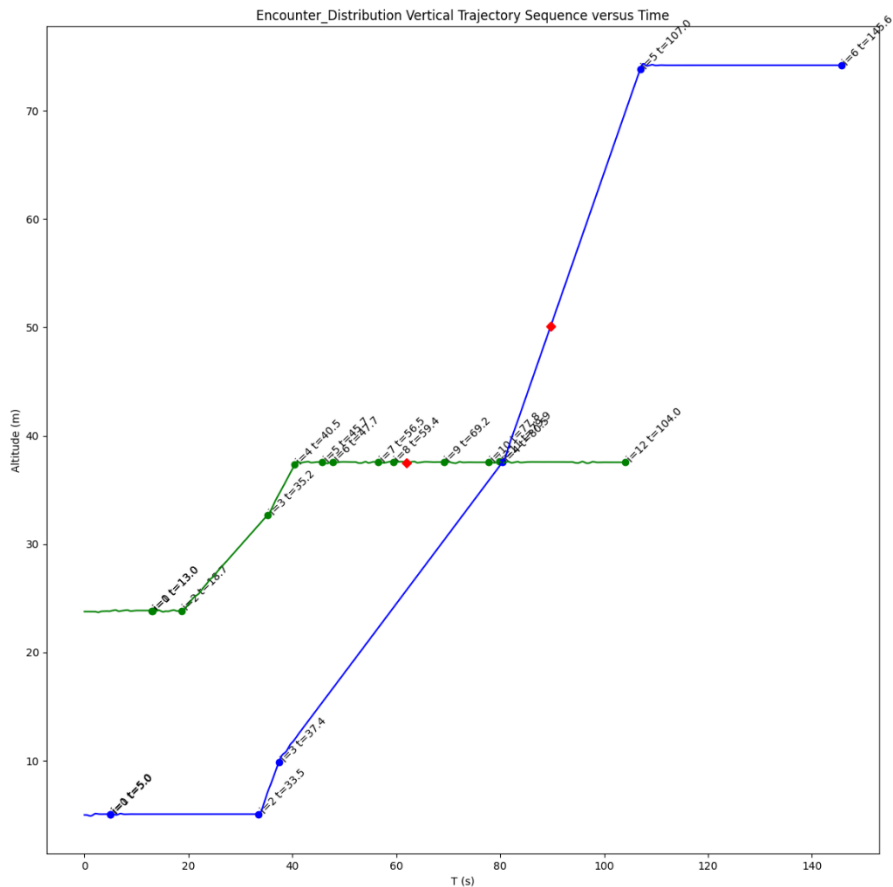


Fig 33 DEG Encounter: Altitude vs Time

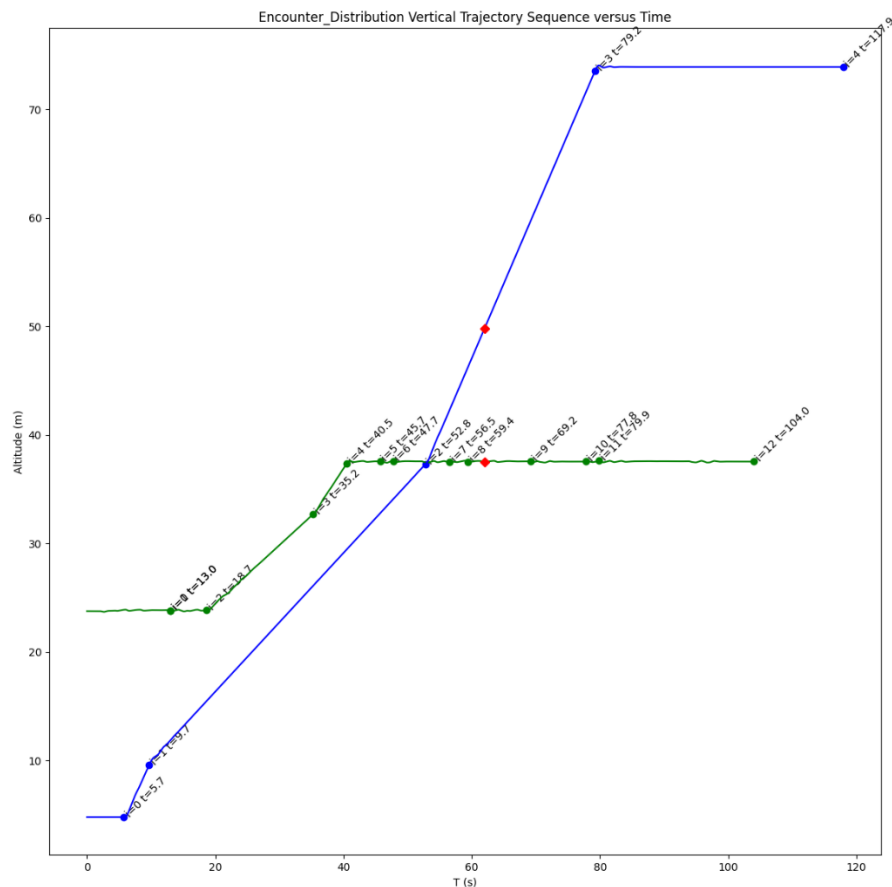


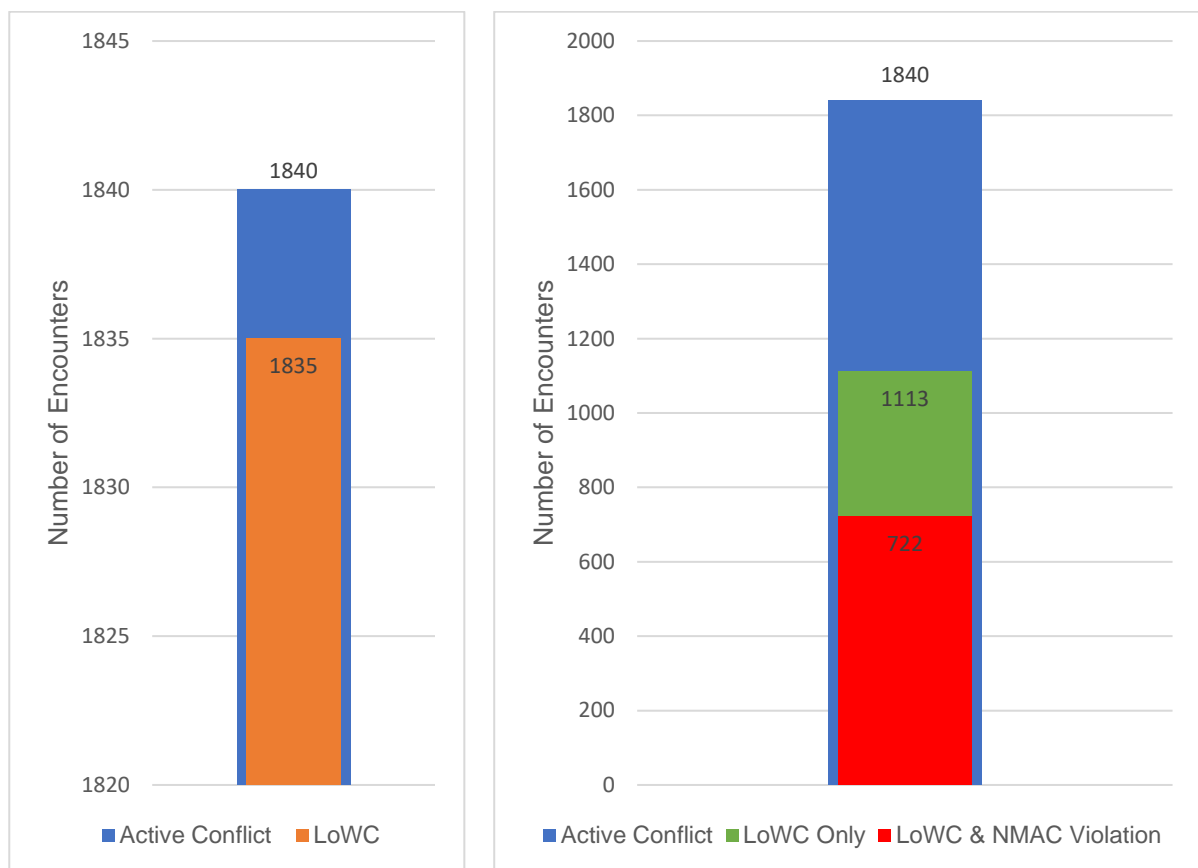
Fig 34 DEG Rotated Encounter: Altitude vs Time

We can see how the sampled CPA conditions are accomplished (VMD = 12,336 m; Ownship Altitude (green line) = 37,470 m; Intruder Altitude (blue line) = 49,806 m).

5.2 Characterizing Unmitigated DEG Encounters

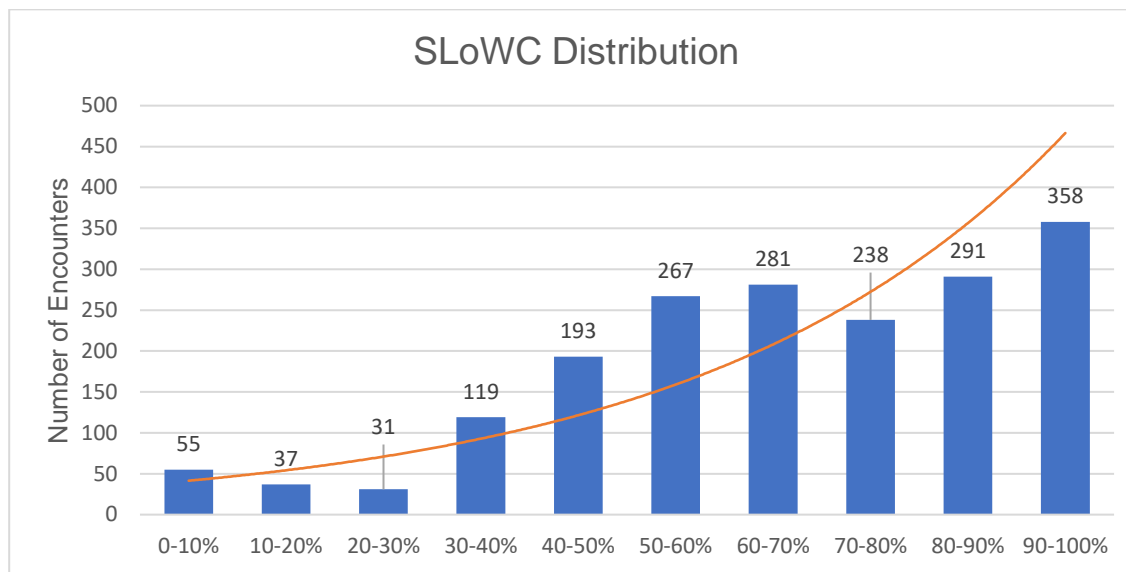
This section shows the rate of LoWC and NMAC violations as well as additional performance metrics calculated using 1870 drone DEG encounters. These encounters are unmitigated in the sense that no dodging manoeuvres have been considered during the simulation. The analysis has been conducted using a tuned version of DAIDALUS v1.0, integrating the well clear detection volumes and alert thresholds proposed in this project. The values of the thresholds are contained in [Table 3.7](#), [Table 3.9](#) and [Table 3.10](#), and the definition of the variables are outlined in [Section 2.3.3](#).

Rate of Losses of Well Clear (LoWC) and NMAC Violations



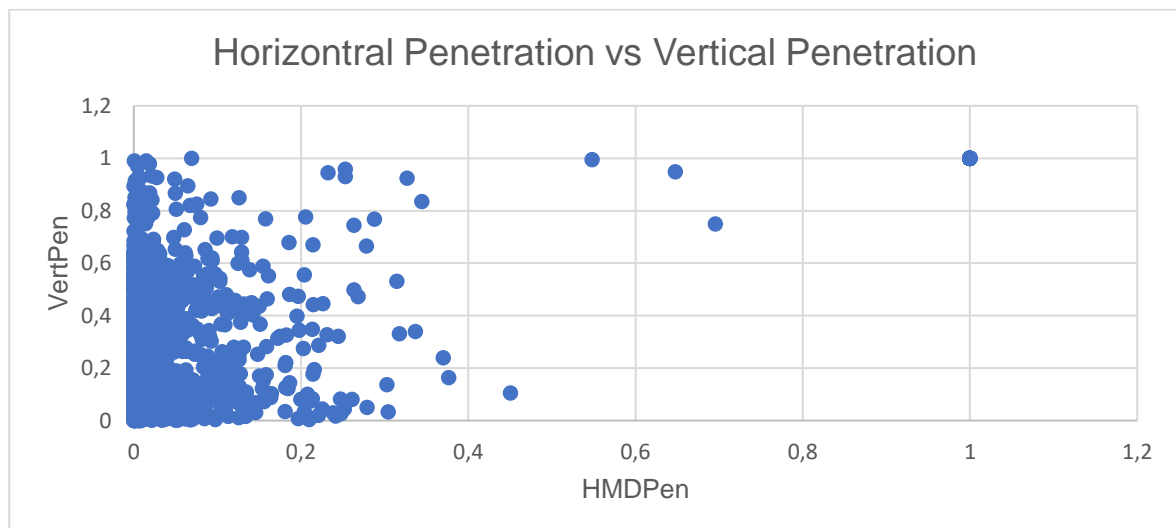
More than 98% of the encounters generated with DEG have triggered an alert during the simulation. From that proportion, 99.7% have violated the Hazard zone (has entered the WCV); 60% of the encounters have lost their Well Clear condition but have not induced a risk of NMAC while 39.2% have violated the NMAC Volume.

Severity of Loss of Well Clear (SLoWC) Distribution



The SLoWC metric evaluates the severity of the local penetration (pen) into all three of the Well Clear Components: Horizontal Proximity, Horizontal Miss Distance and Vertical Separation. Essentially, it captures the most serious instance of LoWC over the whole encounter. The SLoWC metric ranges from 0% indicating Well Clear to 100% representing full penetration (two drones at the same place at the same time). The exponential orange trend line indicates that the majority of DEG encounters have some serious risk of collision.

Horizontal vs Vertical Penetration



The normalized HMD penetration (HMDPen) ranges from 1 to 0, or from the edge of DMOD to full penetration into the DMOD requirement. The normalized Vertical penetration (VertPen) yields a value ranging from 1 to 0, the former indicating the edge of vertical threshold (ZTHR) and the latter representing full vertical penetration into the vertical dimension. Note how DEG encounters have less miss distance in the horizontal domain than in the vertical domain.

CHAPTER 6. CONCLUSIONS & FURTHER IMPROVEMENTS

It has been shown that the DEG tool can be used for the synthetic generation of drone trajectories, simulating future U-Space environments. Despite of the previous, this project recognizes that DEG is still under development. DEG could implement the following improvements to increase the fidelity and the reality in the generation of the trajectories: (1) the tool should be optimized to reduce the refresh rate and thus increase the efficiency and the processing speed. Currently DEG recalculates all flight dynamics 50 times per second (50Hz sample rate). A good improvement would be to implement a 2Hz refresh rate (2) apply joint distributions instead of constraining variables when sampling distribution tables. Basically, if we sample randomly without correlating between variables, we could find situations in which the physical characteristics would not allow the drone models to execute certain objectives and therefore the trajectories would not adjust to reality (3) a wind model should also be implemented to improve the reality of the simulation (4) new drone models should also be added along with its corresponding autopilots (5) predefined manoeuvres or paths could be integrated so that DEG can be used for specific airspace regions.

It is also recognized that DAIUDALUS should be updated to version 2.0 to integrate the dynamic well clear volume logic. To validate and certify DAIDALUS algorithm, several FTS and RTS with HITL should demonstrate the effectiveness of the overall DAA system, both in the air and on the ground, as well as the management and rapport with the ANSPs. In the near future, member states may start collecting drone radar tracks for the benefit and optimization of DAA algorithms.

A complete open-loop evaluation of various WC detection volumes and alert thresholds as well as a comparative analysis regarding the distribution of detection times, relative distances and angles with proposals from other institutions should be performed to highlight the strong points and deficiencies of each one of the strategies.

Several European projects, such as SESARJU's URClearED, are now investigating whether DAA systems will improve or worsen the RP's ability to avoid conflicts. The first findings suggest that by improving pilots' traffic awareness, they will not only have an additional responsibility to monitor nearby traffic, but they will also skew air traffic services' tactical conflict capabilities, particularly those of air traffic controllers.

Despite the global pandemic crisis that occurred during the completion of this project, the author's personal view is that the objectives established at the start were met. This project has expanded his knowledge of air traffic services, developed his research and development capabilities, and improved his data analysis skills.

CHAPTER 7. CLOUD-BASED U-SPACE TACTICAL CONFLICT RESOLUTION SERVICE

This project suggests a prototype implementation for a future U-Space tactical conflict resolution services. In accordance with the CORUS U-Space Concept of Operations, these services could be based on ground facilities (i.e., cloud servers), with all aircraft regularly broadcasting their telemetry (basically, position, speed, and heading) so that DAA algorithms can process them. In fact, onboard sensors will only be used to detect surrounding static obstacles such as buildings and to act as back-up systems in the event of a cloud-server failure. This has many advantages, such as detecting potential conflicts well in advance compared to detecting with on-board sensors, consequently reducing the well clear volume boundaries and increasing the airspace capacity. Cloud servers can execute extensive algorithms with high computational loads in short periods of time, hence reducing the drone's battery consumption. It can also enhance the drone's range and endurance due to a reduction of onboard sensors. Furthermore, cloud servers prevent system saturation in the event of multiple-aircraft encounters while meeting the level of cyber security required by the new U-Space environment.

On the other hand, it remains to be seen how the communications infrastructure will be implemented (i.e., based on current 4G/LTE network) and if the C2 signals will comply with the level of Bit Error Rate (BER) and latency required.

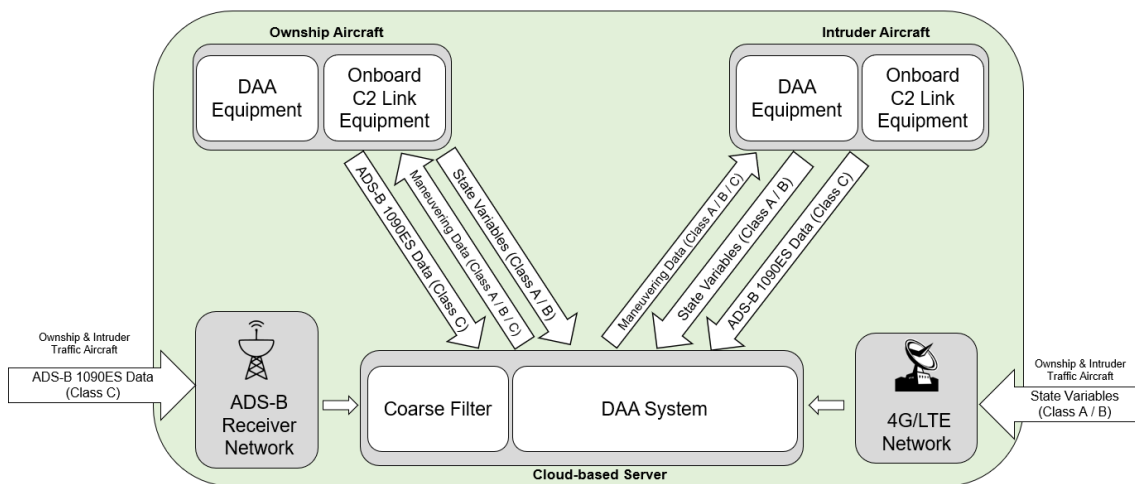


Fig 35 Proposed Cloud-Based Server system overview for autonomous flight operations

7.1 Data Filtering

The proposal of the cloud-based U-Space Tactical Conflict Service of this project considers that only those aircraft that have a real danger of MAC will be provided with DAA capabilities. The rest of the traffic will be ignored until further notice of collision. To do this, all flight trajectories should be filtered so that only pairs (or

multiple pairs) of aircraft with real danger of collision remain tracked by the system (see [Fig 36](#)).

The cloud-based system would receive data from the aircraft periodically (e.g., 500 ms). In each cycle, all reported telemetry would be entered into a coarse filter, to remove unnecessary track pairs from the system, following a Spatial Data Structure (SDS) [20] conflict detection algorithm. Finally, a DAA algorithm would be applied to the remaining trajectories, so that a safe deconfliction manoeuvre would be issued, ensuring the integrity of the airspace.

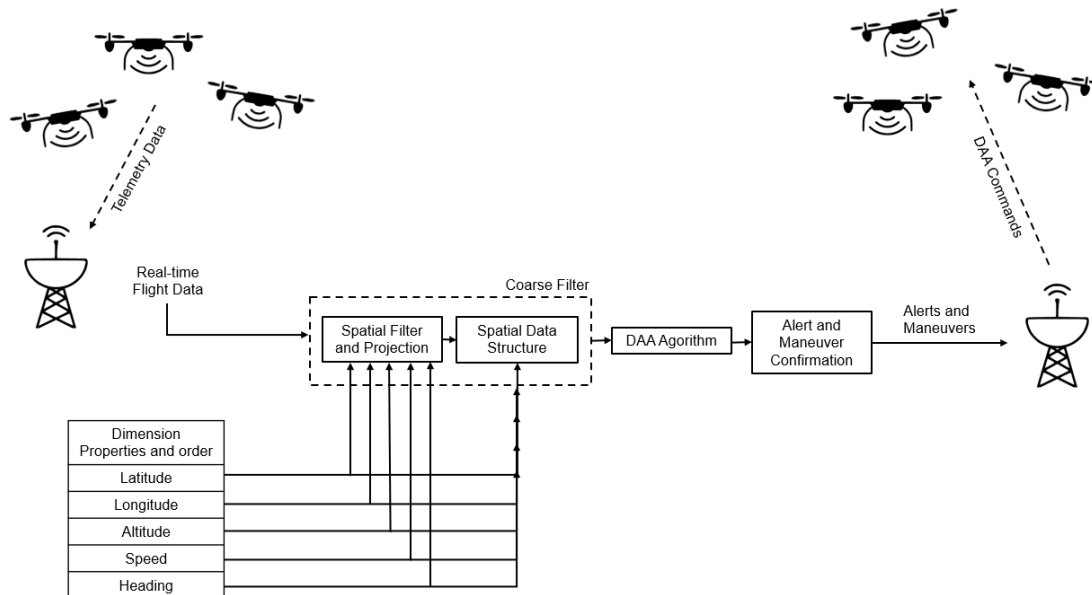


Fig 36 Cloud-based System Processing Overview

7.1.1 Coarse Filter

Pairs of aircraft tracks who have not conflicted are discriminated at this stage and thus much unnecessary processing is avoided, saving computational load. The coarse filter takes the current real-time flight telemetry and calculates whether the aircraft could potentially come into conflict within a certain prediction time with the surrounding traffic.

For a pair of users to pass the coarse filter, a potential conflict must be detected by an SDS conflict detection algorithm. Here, the state variables (latitude, longitude, altitude, speed and heading) of all the aircraft flying in the VLL airspace are sent to ground based servers via ADS-B ground receivers or 4G/LTE networks. All this raw data is then inserted into the Spatial Filter. This filter checks whether the dimension properties and orders of magnitude are coherent, applying the Spatial Data equation defined in [Section 7.1.1.1](#). Then each object (user) is stored as a table in a dynamic database. These objects contain the minimum required data to coherently store the reported data as well as a pointer to link previous with updated objects. A hash table links the unique identifier with the last updated data. Each user is then "placed" in carefully designed cells to maximize the probability of conflict detection while minimizing the system's

memory usage. Conflicts with neighbouring users are analysed. The ones who do not conflict will be discarded (if they are separated by more than one cell, it is said that they are not in conflict). Therefore, users who are likely to conflict go to the next step where a DAA algorithm will predict if a LoWC will occur and if so, the user will be notified of the corresponding alert level and deconfliction manoeuvre. In the case of completely autonomous drone operations, the ground-based system will issue automatic evasive manoeuvres.

7.1.1.1 *Spatial Data Equation*

The spatial data equation will serve as an error mitigator for the actual flight data. The use of relatively cheap on-board sensors in environments with high levels of electromagnetic radiation can result in an increased noise at the input of the sensors. In addition, Global Navigation Satellite Systems (GNSS) that provide the location of the drones also have some level of uncertainty. Equation (7.1) weights the last three reported data for latitude, longitude, altitude, speed, and heading as follows:

$$x_{out} = 0.8x_t + 0.1x_{t-1} + 0.1x_{t-2} \quad (7.1)$$

where:

x_{out} is the resulting average input data and,
 x_{t-i} is the input data for each time interval.

BIBLIOGRAPHY

- [1] SESAR JU, "European ATM Master Plan: Roadmap for the safe integration of drones into all classes of airspace," 19 March 2018. [Online]. Available: <https://www.sesarju.eu/sites/default/files/documents/reports/European%20ATM%20Master%20Plan%20Drone%20roadmap.pdf>.
- [2] CORUS Consortium, "U-space Concept of Operations Vol2," 2019. [Online]. Available: <https://www.sesarju.eu/sites/default/files/documents/u-space/CORUS%20ConOps%20vol2.pdf>.
- [3] EU/EASA, "COMMISSION IMPLEMENTING REGULATION (EU) No 923/2012," 2012. [Online]. Available: <https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:02012R0923-20171012&from=EN>.
- [4] SESAR JU, "U-space Blueprint brochure final," 2017. [Online]. Available: <https://www.sesarju.eu/sites/default/files/documents/reports/U-space%20Blueprint%20brochure%20final.PDF>.
- [5] FAA and EUROCONTROL, "Principles of Operations for the Use of ASAS," 19 6 2001. [Online]. Available: <https://www.eurocontrol.int/care-asas/gallery/content/public/docs/po-asas71.pdf>.
- [6] EASA, "Easy Access Rules for Standardised European Rules of the Air (SERA)," 2020. [Online]. Available: <https://www.easa.europa.eu/sites/default/files/dfu/Easy%20Access%20Rules%20for%20Standardised%20European%20Rules%20of%20the%20Air%20%28SERA%29.pdf>.
- [7] ICAO, "Rules of the Air – Annex 2," 2005. [Online]. Available: https://www.icao.int/Meetings/anconf12/Document%20Archive/an02_cons%5B1%5D.pdf.
- [8] EUROCONTROL and EASA, "UAS ATM Flight Rules," 16 5 2019. [Online]. Available: <https://www.eurocontrol.int/publication/uas-atm-flight-rules>.
- [9] EUROCONTROL, "UAS ATM Integration," 2018. [Online]. Available: <https://www.eurocontrol.int/sites/default/files/publication/files/uas-atm-integration-operational-concept-v1.0-release%2020181128.pdf>.
- [10] ICAO, "Manual on Remotely Piloted Aircraft Systems (RPAS)," 2015. [Online]. Available: <https://skybrary.aero/bookshelf/books/4053.pdf>.
- [11] R. DO-365A, "Minimum Operational Performance Standards for Detect and Avoid Systems," 2020.
- [12] FAA, "Coordination Between Airborne Collision Avoidance Systems," 2013.
- [13] M. Johnson, C. Santiago and E. Mueller, "Characteristics of a Well Clear Definition and Alerting Criteria for Encounters between UAS and Manned Aircraft in Class E Airspace," 2015. [Online]. Available: <https://ntrs.nasa.gov/citations/20190027490>.
- [14] EUROCAE, "Operational Services and Environment Description For Detect and Avoid In Class D-G Airspaces under IFR/VFR," 2019.
- [15] M. Kochenderfer, D. Griffith and J. Olszta, "On Estimating Mid-Air Collision Risk," 2010. [Online]. Available: <https://arc.aiaa.org/doi/pdf/10.2514/6.2010-9333>.
- [16] A. Weinert, L. Alvarez, M. Owen and B. Zintak, "A Quantitatively Derived NMAC Analog for Smaller Unmanned Aircraft Systems Based on

- Unmitigated Collision Risk,” 2020. [Online]. Available: <https://www.preprints.org/manuscript/202011.0503/v1>.
- [17] E. Lester and A. Weinert, “Three Quantitative Means to Remain Well Clear for Small UAS in the Terminal Area,” 2019. [Online]. Available: <https://ieeexplore.ieee.org/document/8735171>.
- [18] A. Narkawicz, C. Munoz and A. Duple, “Sensor Uncertainty Mitigation and Dynamic Well Clear Volumes in DAIDALUS,” 2018. [Online]. Available: <https://shemesh.larc.nasa.gov/fm/papers/DASC2018-DAIDALUS-draft.pdf>.
- [19] E. Mueller, C. Santiago and S. Watzka, “Piloted “Well Clear” Performance Evaluation of Detect-and-Avoid Systems with Suggestive Guidance,” 2016. [Online]. Available: <https://ntrs.nasa.gov/api/citations/20160006632/downloads/20160006632.pdf>.
- [20] M. Perez Batlle, R. Cuadrado and E. Pastor, “An Efficient Strategic-Level Conflict Detection Algorithm Based on Spatial Data Structures”.

APPENDIX

TITLE: Evaluation of Remain Well Clear and Collision Avoidance for Drones

DEGREE: Bachelor's degree in Aerospace Systems Engineering

AUTHOR: Santiago L. del Hierro Mosteiro

DIRECTOR: Enric Pastor Llorens

DATE: October 19th , 2021

APPENDIX A. DYNAMIC WELL CLEAR VOLUME DEMONSTRATION

This section demonstrates that the Hazard Alert Zone should be dynamically adjusted over the course of a conflict depending on the aircraft performances. The following investigation has been conducted using a dedicated MATLAB tool called the Hazard Volume Analyzer tool.

The simulations have taken into account different collision courses (approaching head-on, converging and overtaking) with relative approach angles of 180, 45 and 0 degrees respectively. Only the most critical scenarios have been executed, where both aircraft have the highest relative speed. That is, the fastest aircraft flying at top speed and the slowest at stall speed. For simplicity, stall speeds are considered one third of the maximum speeds exposed in [Table 3.4](#).

Two additional fixed parameters were considered: NMAC volume limits as specified in [Table 3.5](#) and intruder delay system (sensor processing time plus pilot reaction time) set to 7 seconds as shown in [Table 3.6](#).

The Hazard Volume Analyzer tool works as follows: first, both ownship and intruder aircraft are placed in the initial position at co-altitude (for simplicity (0,0,0) m in cartesian coordinates). Then the intruder is placed in the surrounding positions spaced a time dt along the x-axis (aircraft longitudinal axis) and a time t along the y-axis (aircraft lateral axis). For each position and collision course, the WC status is calculated, using the HAZ violation logic. Different colours denote different conflict status of the intruder with respect to the ownship (green: Clear of Conflict (CoC); black: WCV violation; red: NMACv violation). For simplicity, all the encounters involved two aircraft flying in cruise phase and only horizontal dodging manoeuvres were considered.

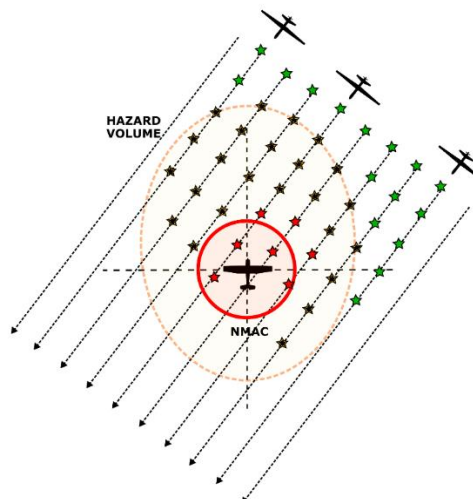


Fig 37 Conflict status according to intruder position

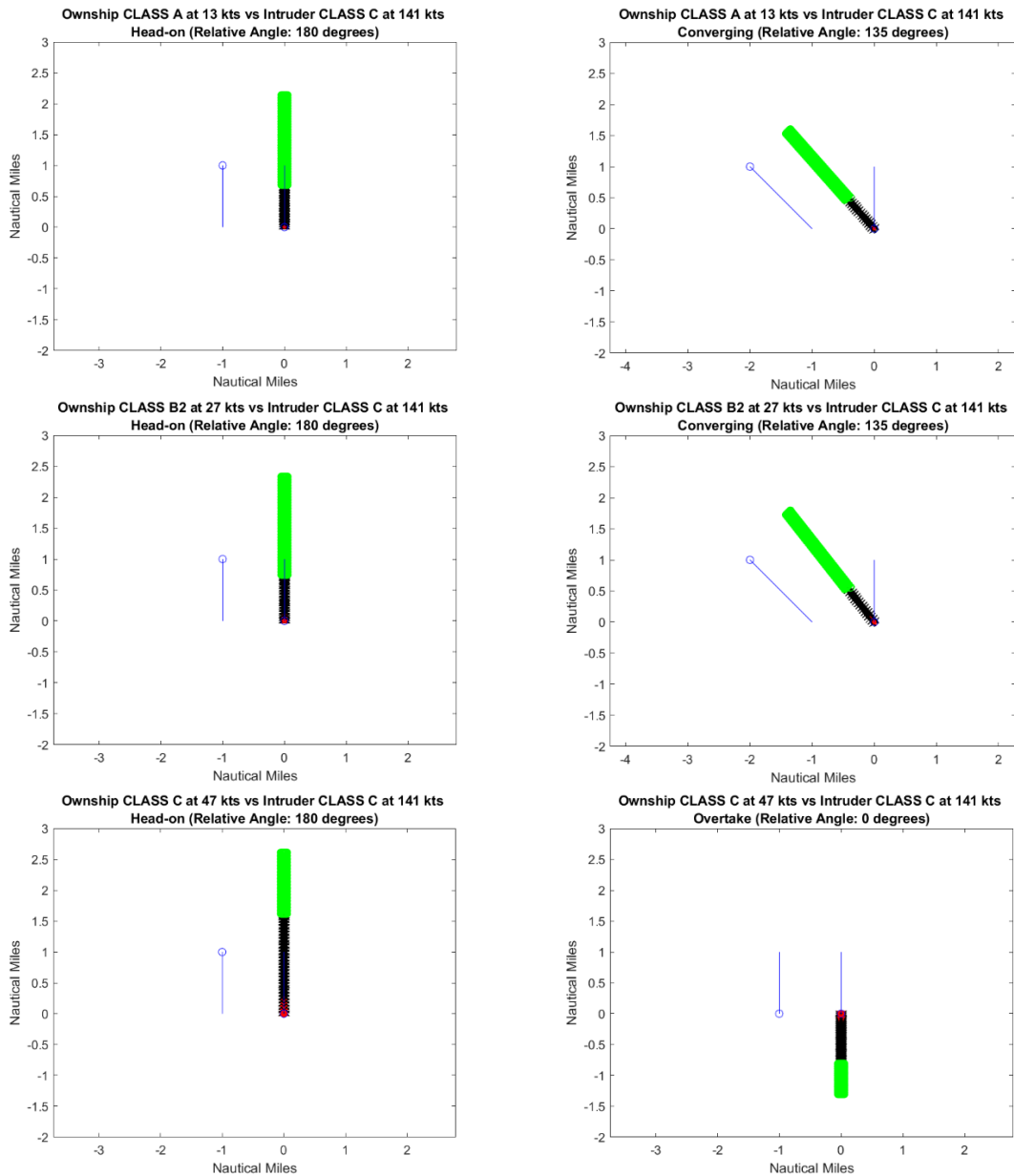


Fig 38 Generated heatmap plot examples

Fundamentally, the greater the aircraft is the larger WCV should be designated (see [Fig 38](#)). That is, to cope with greater maximum speeds and less manoeuvrable capabilities. Furthermore, greater WCVs shall be assigned to manned aircraft due to the inability to visually detect surrounding drones (in the case Class C aircraft are not mandated to be equipped with proper conspicuity devices).

APPENDIX B. COMPREHENSIVE DESCRIPTION OF DAIDALUS ALGORITHM

The following subparagraphs exposes a comprehensive explanation of the main functionalities of the DAIDALUS algorithm. The values of the thresholds are shown in [Table 3.7](#), [Table 3.9](#) and [Table 3.10](#), and the definition of the variables can be found in Section [2.3.3](#). The code is written in PYTHON language.

B.1 Well Clear Logic

The Boolean function *check_hazard_zone* implements the well-clear logic. The relative position and velocity of the aircraft are inputs to this function. If the aircraft are in LoWC in the current time, the method returns TRUE. This condition is checked at the start of the processing logic to see if both aircraft are in LoWC at the start of the encounter. Two conditions in the horizontal domain and one in the vertical domain are evaluated by the function.

Well Clear Violation:

If the horizontal and vertical well clear violations are TRUE then there is a LoWC.

$$\text{RWC_out.WCV} \equiv \text{RWC_out.WCVxy and RWC_out.WCVz}$$

Horizontal Well Clear Violation:

If the current horizontal range between aircraft (r) is less or equal than the horizontal size of the HAZ (S^*) and the predicted horizontal miss distance (HMD_p) is less or equal than the distance modification (DMOD) then the violation of the well clear status in the horizontal domain is TRUE.

$$\text{RWC_out.WCVxy} \equiv r \leq S^* \text{ and } \text{HMD}_p \leq \text{DMOD}$$

Vertical Well Clear Violation:

If the current vertical separation (d_h) is less or equal than the vertical separation threshold (ZTHR) then there is a vertical LoWC.

$$\text{RWC_out.WCVz} \equiv d_h \leq \text{ZTHR}$$

B.2 Detection Logic

If both aircraft are well clear at the beginning of the encounter, then the *WCV_interval* function will determine whether the intruder aircraft will lose its well clear status in relation to the ownship aircraft within a lookahead time. The status of both aircraft, as well as a lookahead time interval $[B, T]$ are all inputs to this

function. A time interval $[t_{in}, t_{out}]$ within $[B, T]$ is returned by this method. If $t_{in} \leq t_{out}$, then t_{in} denotes the time to start the well-clear violation and t_{out} denotes the time to exit the well-clear violation, assuming constant velocity (linear) projections of the aircraft current states. If the aircraft are not predicted to be in LoWC within the interval $[B, T]$ then the returned time interval is empty or non-coherent, i.e., $t_{in} > t_{out}$. The value of B is set to 0s whilst T is 200s.

Vertical Detection: *vertical_WCV_interval* function

If the relative vertical speed (v_z) is null and the absolute value of the current vertical separation (s_z) is less or equal than the vertical separation threshold (ZTHR) then it is returned the time interval $[B, T]$.

```
if (math.isclose(vz, 0) and abs(sz) <= rwc_param.ZTHR):
    return itv.Interval(B, T)
```

If the vertical speed (v_z) is null then it is returned a non-coherent time interval $[T, B]$.

```
if (math.isclose(vz, 0)):
    time_in = T
    time_out = B
    return itv.Interval(time_in, time_out)
```

Next it is computed the entry t_{in} and exit t_{out} time when the intruder intersects a rectangle of height ZTHR (assuming TCOA = 0s).

```
act_H = max(rwc_param.ZTHR, abs(vz) * rwc_param.TCOA)
```

$$\left[-\frac{\text{sign}(v_z) \cdot \text{ZTHR} - s_z}{v_z}, \frac{\text{sign}(v_z) \cdot \text{ZTHR} - s_z}{v_z} \right]$$

From the previous output, if t_{in} is greater than the lookahead time T or t_{out} is smaller than the initial time interval B then there is an incongruence so the returned interval is $[T, B]$.

```
if (T < t_in or t_out < B):
    time_in = T
    time_out = B
    return itv.Interval(time_in, time_out)
```

Otherwise, the returned interval is the maximum between the in and out value or the given lookahead time interval.

```

time_in = max(B, tin)
time_out = min(T, tout)
return itv.Interval(time_in, time_out)

```

In the case the returned values are equal ($t_{in} = t_{out}$) and the predicted horizontal well clear violation ($RWC_out.WCVxy$) at the intruder's projected position is TRUE then the LoWC is happening at the perimeter of the WCV (tangent to the parametric line) and the returned interval is $[t_{in}, t_{in}]$.

```

step = s2 + tin * v2

if (horizontal_WCV(step, v2, rwc_param)):
    time_in = tin
    time_out = tout
    return itv.Interval(time_in, time_out)

```

The following logic will offer a similar rationale for the horizontal domain once the well clear condition has been confirmed in the vertical domain.

Horizontal Detection: *horizontal_WCV_interval* function

First, the parameters a , b and c of the quadratic equation (i.e., $ax^2 + bx + c = 0$) that forms the parabolic perimeter of the well clear volume are declared. Note that the actual shape of the WCV depends on the aircraft states.

```

a = v2
b = 2 (s*v) + TAUMOD * v2
c = s2 + TAUMOD(s*v) – DMOD2

```

If the scalar product of the velocity (a) equals 0 and the norm of the current horizontal separation ($\|s\|$) is less or equal the modified distance (DMOD) then the returned time interval ranges from 0 to lookahead time T .

```

if a = 0 and  $\|s\| \leq DMOD$ 
    time_in = 0
    time_out = T
    return itv.Interval(time_in, time_out)

```

If $\|s\|$ is less or equal than DMOD it means there is a LoWC so the output interval will range from 0 to the minimum value between the lookahead time (T) and the time the intruder exits the WCV.

If the scalar product between the relative distance (s) and the relative speed (v) is greater or equal than 0 or the discriminant of the quadratic function ($b^2 - 4ac$) is less than 0, it means the projected intruder's trajectory will not intersect with the parametric line of the hazard zone (or WCV perimeter) of the ownship aircraft. Hence, the returned time interval is a nonsense $[T,0]$.

If the above conditions did not provide a result, the next processing logic will try to identify the first and last time the intruder is in LoWC with the ownship. For that, it is calculated the roots of the quadratic function of the WCV perimeter. If the discriminant is equal to zero it means the intruder will only intersect the WCV in one point. If the discriminant is greater than 0 then the intruder will entry the hazard volume and exit in two different points. The correspondent time values will be the output of this logic.

```
t = (-b - math.sqrt(discr)) / (2 * a)
if (horizontal.Delta(s, v, rwc_param.DMOD) >= 0 and t <= T):
    time_in = max(0, t)
    time_out = min(T, horizontal.Theta_D(s, v, 1, rwc_param.DMOD))
return itv.Interval(time_in, time_out)
```

APPENDIX C. HORIZONTAL AND VERTICAL DRONE PERFORMANCE TEST

C.1 Horizontal and Vertical Performance Test – DJI F450

The following subparagraphs explain a series of tests that have allowed tuning the parameters of the PID controllers of the two drones available in DEG. The tests contain several stages, each designed to calculate the horizontal and vertical performance of the vehicle.

The main objective of the tests is to calculate the operational envelope of the two drones. That is, to compute the maximum speeds, accelerations, decelerations, climb and descent rates.

The default horizontal and vertical performance test consist of four stages. The first stage is a straight-line horizontal trajectory where the drone is flown for 50 meters, without variation in altitude. The second stage is a climb followed by a descent, both with an inclination with respect to the horizontal plane of 45 degrees and same distance flown as in the previous straight line. The third stage consists of a completely vertical climb and subsequent descent, being the distance flown of 50 meters. The fourth and final stage is a straight-line path flown without altitude variation. The same test is carried out at different initial altitudes, segment lengths and slope inclinations.

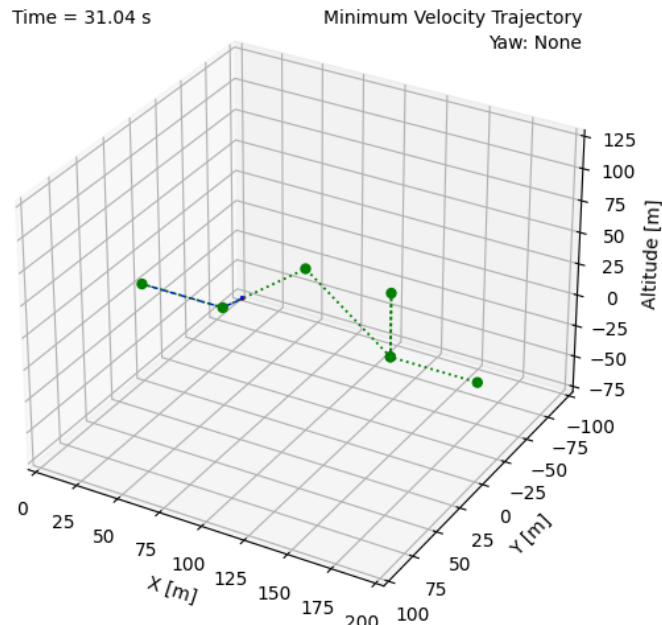
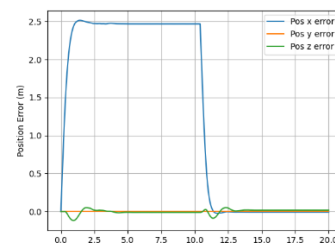
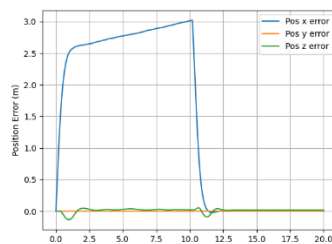
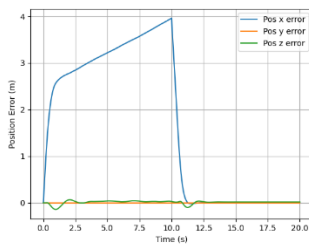
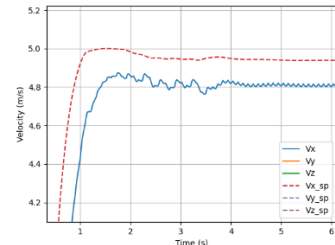
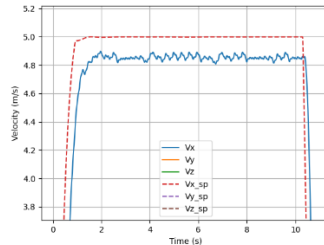
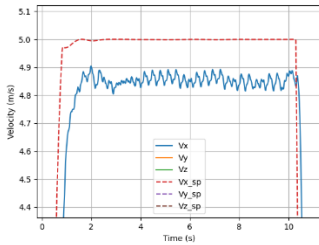


Fig 39 Horizontal and Vertical Performance Test Scenario

Stage 1 - Segment 1: From initial position to Waypoint 1

- As can be observed in the graphs below, the drone is not able to reach a speed of 5 m/s. The maximum sustained speed of the DJI F450 when

flying straight and level is $v_{\max} = 4.8 \text{ m/s}$. Note that the position error for the x-axis (graphs on the lower level) increases when flying at speeds greater than the maximum. In addition, see how the amplitude of the speed gradually decreases until it reaches the equilibrium. In consequence, the system is said to be underdamped.

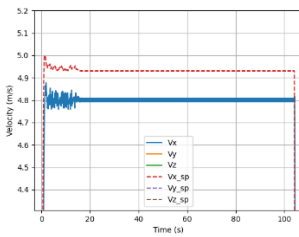


Speed and Position Error vs Time
($v_{\text{desired}}^{\text{mean}} = 5 \frac{\text{m}}{\text{s}}$)

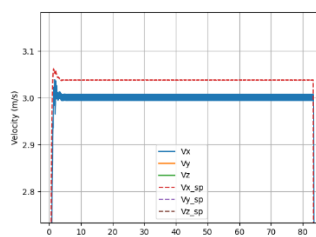
Speed and Position Error vs Time
($v_{\text{desired}}^{\text{mean}} = 4.9 \frac{\text{m}}{\text{s}}$)

Speed and Position Error vs Time
($v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}$)

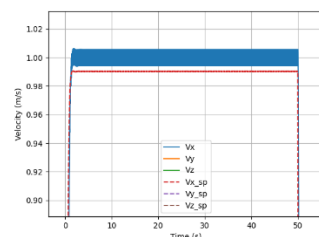
- There is no degradation in performance due to variations in altitude or periods of time (when flying straight and level flight). The drone can be sustained continuously at a speed less or equal the maximum at any altitude and for an unlimited time.



Speed vs Time
($v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}$; $\text{Altitude} = 1000\text{m}$; $\text{Distance} = 500\text{m}$)

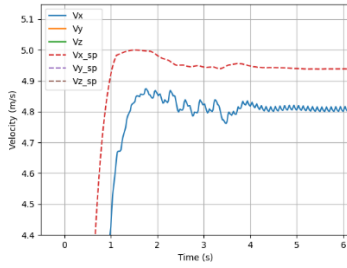


Speed vs Time
($v_{\text{desired}}^{\text{mean}} = 3 \frac{\text{m}}{\text{s}}$; $\text{Altitude} = 500\text{m}$; $\text{Distance} = 250\text{m}$)

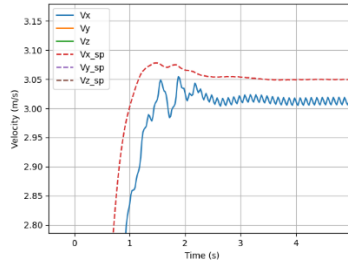


Speed vs Time
($v_{\text{desired}}^{\text{mean}} = 1 \frac{\text{m}}{\text{s}}$; $\text{Altitude} = 200\text{m}$; $\text{Distance} = 50\text{m}$)

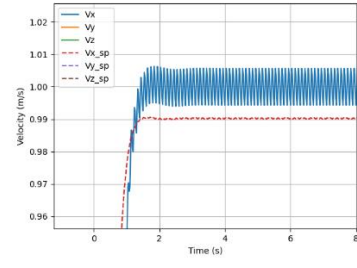
- The greater the speed is, the more position error and longer transitory regime. The position error remains constant for any segment length.



Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}; \text{distance} = 500\text{m})$



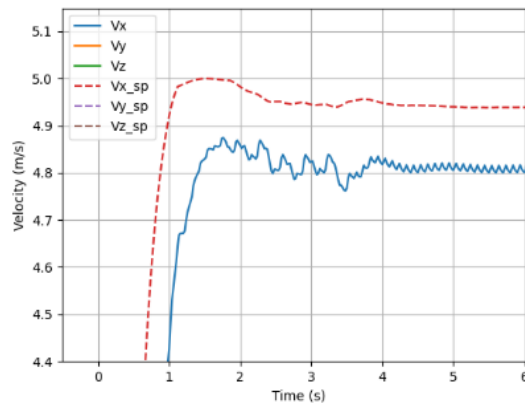
Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 3 \frac{\text{m}}{\text{s}}; \text{distance} = 50\text{m})$



Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 1 \frac{\text{m}}{\text{s}}; \text{distance} = 50\text{m})$

- The drone can accelerate from stationary to a stable maximum speed in $t = 4.2$ s. This gives a maximum acceleration of:

$$a_{\text{max}} = \frac{4.8 \frac{\text{m}}{\text{s}}}{4.2 \text{s}} = 1.14 \frac{\text{m}}{\text{s}^2}$$



Speed vs Time
 $(v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}})$

Stage 2 – Segment 2 and 3: From Waypoint 1 to Waypoint 3

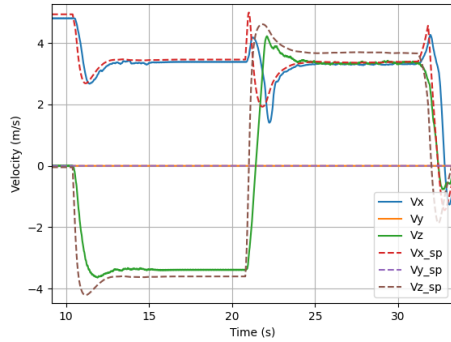
* The speeds of the plots are inversely dimensioned (negative values when climbing).

- The drone is almost able to reach a maximum speed of $v_{\text{max}} = 4.8$ m/s when climbing and descending with a path angle of 45 degrees. Due to the transitioning waypoints, the drone will fly at a smaller maximum speed. If we calculate the module of the speed vector for a maximum speed of 4.8 m/s, we obtain:

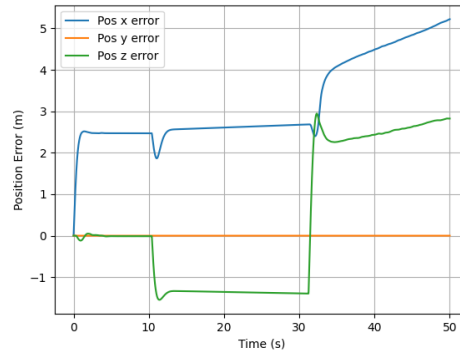
$$|v_{\text{max}}| = \sqrt{v_x^2 + v_z^2} = \sqrt{3.38^2 \frac{\text{m}}{\text{s}} + 3.38^2 \frac{\text{m}}{\text{s}}} = 4.8 \frac{\text{m}}{\text{s}}.$$

Observe the green and blue lines of the plot below corresponding to the vertical and horizontal speeds. These speeds are slightly less than 3.38

m/s which means that it is not possible to reach a maximum speed of 4.8 m/s. Note in the following graph how the trend of the position error is increasing. The descent phase presents a higher rate of position error than the climb phase over time.

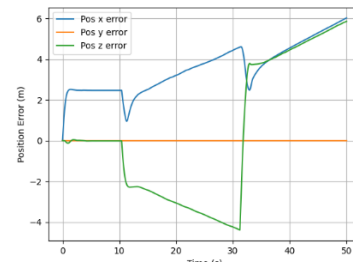
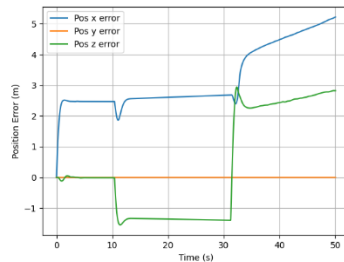
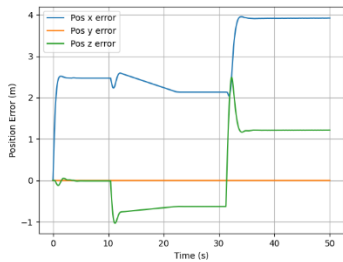
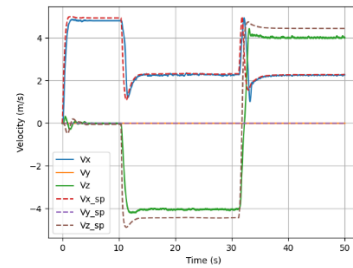
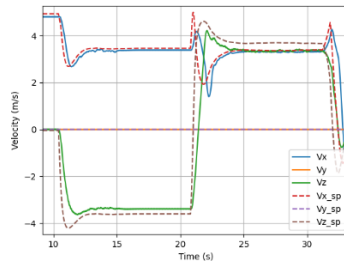
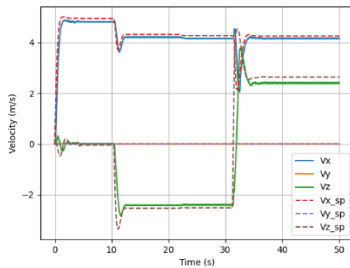


Speed vs Time
 $(v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}; \text{distance} = 50\text{m per segment}; \text{inclination} = 45^\circ)$



Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}; \text{distance} = 100\text{m per segment}; \text{inclination} = 45^\circ)$

- The steeper the slope is, the greater position error and smaller maximum speed.

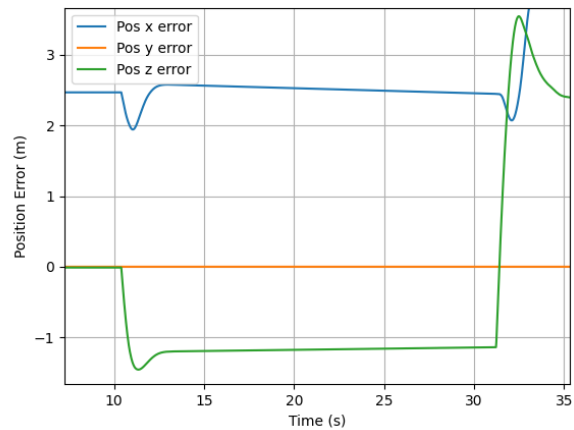
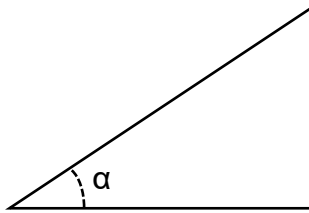


Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}; \text{distance} = 100\text{m per segment}; \text{inclination} = 30^\circ)$

Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}; \text{distance} = 100\text{m per segment}; \text{inclination} = 45^\circ)$

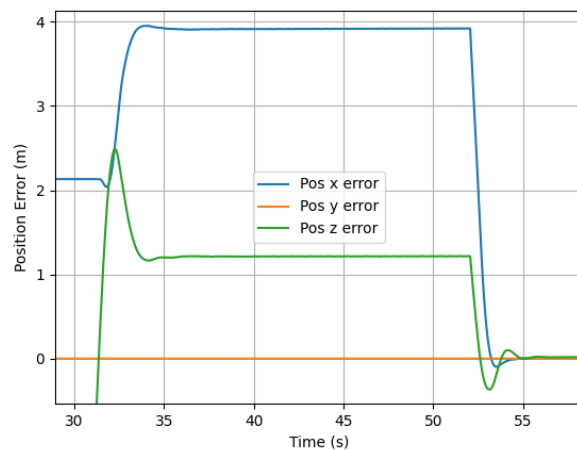
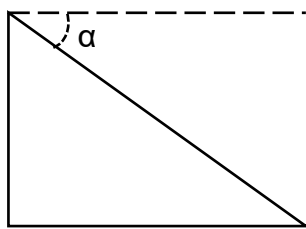
Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}; \text{distance} = 100\text{m per segment}; \text{inclination} = 60^\circ)$

- If the path angle is $\alpha > 42.5$ degrees, the drone will not be able to reach the maximum speed of 4.8 m/s when climbing.



($v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}$; distance = 100m per segment; inclination = 42.5°)

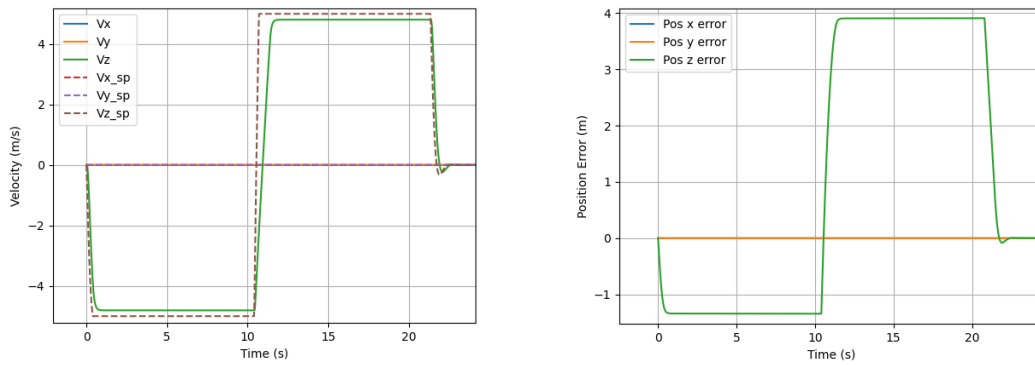
In opposite, the drone will not be able to reach the maximum speed of 4.8 m/s when descending if the path angle is $\alpha > 30$ degrees.



($v_{\text{desired}}^{\text{mean}} = 4.8 \frac{\text{m}}{\text{s}}$; distance = 100m per segment; inclination = 30°)

Stage 3 – Segment 4 and 5: From Waypoint 3 to Waypoint 4 to Waypoint 3

- The drone is able to reach the maximum speed of 4.8 m/s when climbing and descending vertically (90 degrees with respect to the horizontal plane). But because the drone has an initial position error, the predicted values of position and speed will be distorted. Note that the position error is greater in the second segment due to the inaccurate prediction of position (the drone cannot overcome the delay because is already flying at full speed).

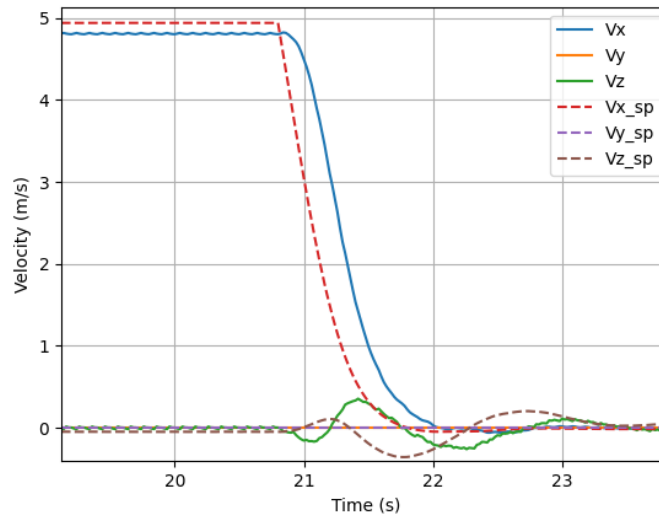


Speed and Position Error vs Time ($v_{desired}^{mean} = 4.8 \frac{m}{s}$; $distance = 50m$)

Stage 4 – Segment 6: From Waypoint 3 to Waypoint 5

- The drone can decelerate from a stable maximum speed to 0 m/s in $t = 22s - 20.8s = 1.2s$. This gives a maximum braking deceleration of

$$a_{max} = \frac{-4.8 \frac{m}{s}}{1.2s} = -4 \frac{m}{s^2}$$



Speed vs Time ($v_{desired}^{mean} = 4.8 \frac{m}{s}$; $distance = 100m$)

C.2 Horizontal and Vertical Performance Test – DJI F450 FAST

The same horizontal and vertical type of test has been carried out with the higher performance drone DJI F450 FAST. In this case, the segments are enlarged to allow stabilization of the drone when transitioning between waypoints.

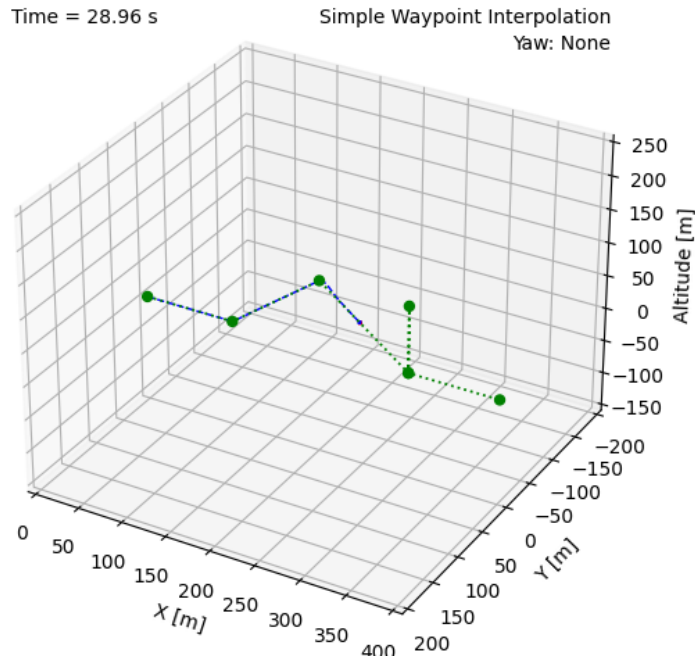
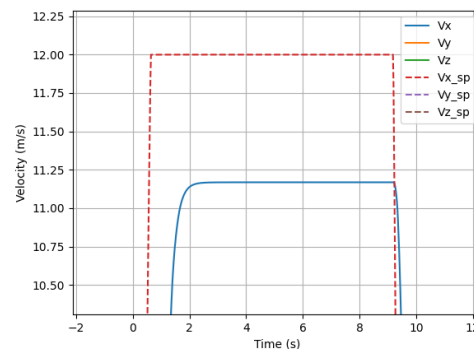
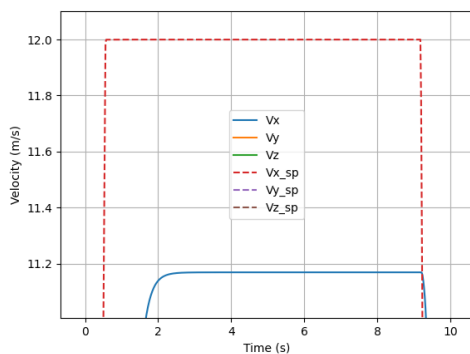
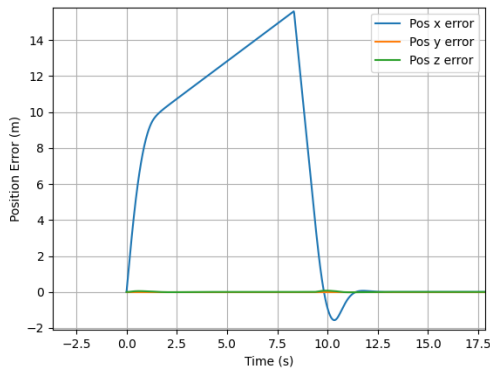


Fig 40 Horizontal and Vertical Performance Test Scenario

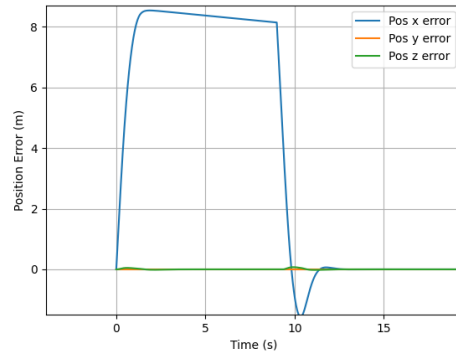
Stage 1 - Segment 1: From initial position to Waypoint 1

- Note how the trend of the x-axis position error (graphs on the lower level) increases over time when the drone is asked to fly at a higher speed than the effective maximum. The maximum sustained speed of the DJI F450 FAST when flying straight and level is $v_{\max} = 11.1 \text{ m/s}$. This drone has a greater initial position error since its heavier and less responsive than the smaller DJI F450.



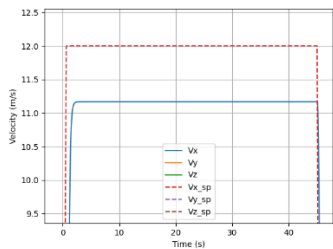


Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 12 \frac{\text{m}}{\text{s}}; \text{Distance} = 100\text{m})$

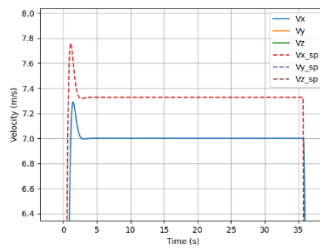


Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}; \text{Distance} = 100\text{m})$

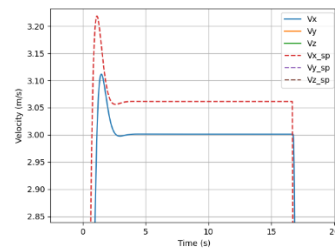
- There is no performance degradation due to variations in altitude or periods of time (when flying straight and level flight). The drone can be sustained continuously at a speed less or equal the maximum at any altitude and for an unlimited time. In addition, this drone has no overshoot in speed when flying at maximum speed (critically damped system) but does when flying below its maximum speed (underdamped system).



Speed vs Time
 $(v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}; \text{Altitude} = 1000\text{m}; \text{Distance} = 500\text{m})$

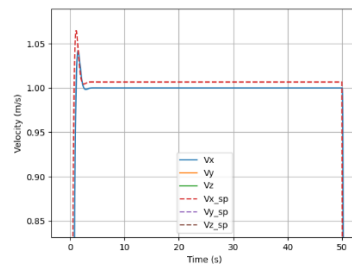
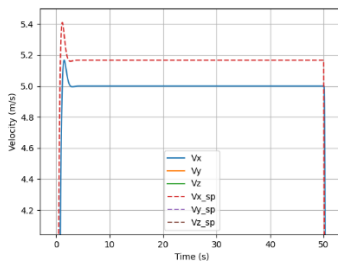
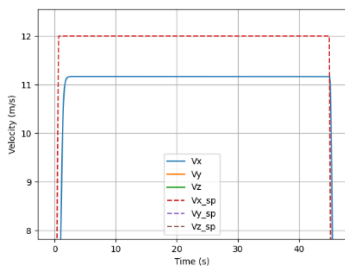


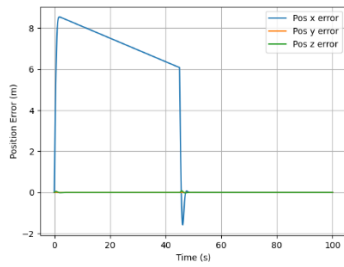
Speed vs Time
 $(v_{\text{desired}}^{\text{mean}} = 7 \frac{\text{m}}{\text{s}}; \text{Altitude} = 500\text{m}; \text{Distance} = 250\text{m})$



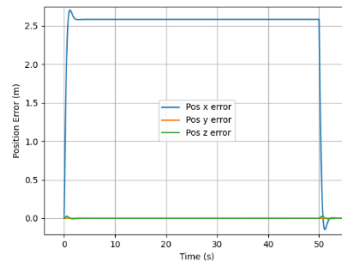
Speed vs Time
 $(v_{\text{desired}}^{\text{mean}} = 3 \frac{\text{m}}{\text{s}}; \text{Altitude} = 200\text{m}; \text{Distance} = 50\text{m})$

- The greater the speed is, the more position error and higher overshoot. The position error remains constant for any segment length except when the drone flies at the maximum speed of 11.1 m/s.

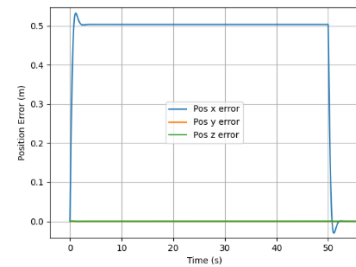




Speed and Position Error vs Time
($v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}$; distance = 500m)



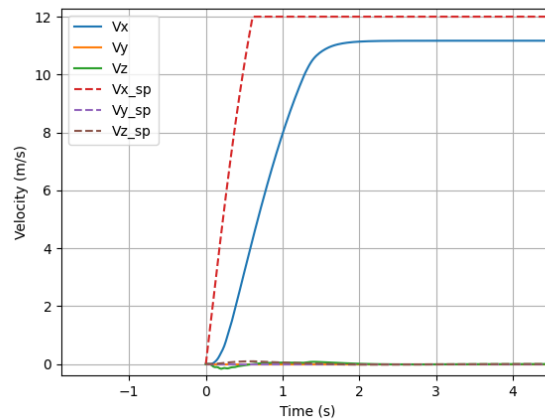
Speed and Position Error vs Time
($v_{\text{desired}}^{\text{mean}} = 5 \frac{\text{m}}{\text{s}}$; distance = 250m)



Speed and Position Error vs Time
($v_{\text{desired}}^{\text{mean}} = 1 \frac{\text{m}}{\text{s}}$; distance = 50m)

- The drone accelerates from stationary to a stable maximum speed in $t = 2.1$ s. This gives a maximum acceleration of:

$$a_{\text{max}} = \frac{11.1 \frac{\text{m}}{\text{s}}}{2.1 \text{s}} = 5.28 \frac{\text{m}}{\text{s}^2}$$

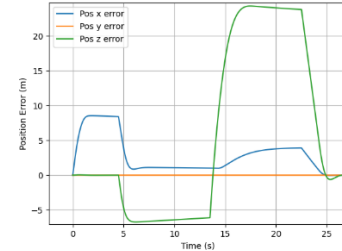
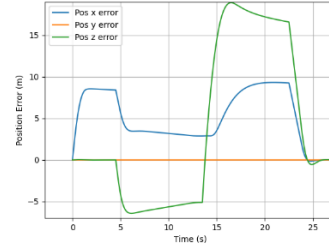
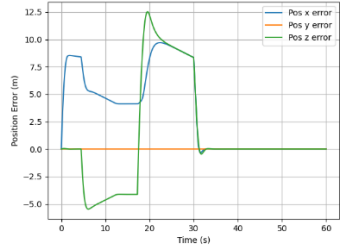
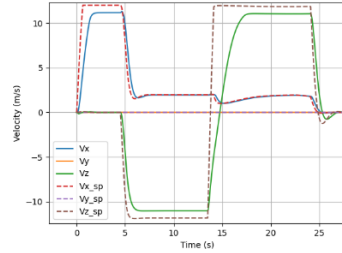
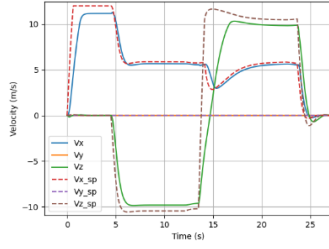
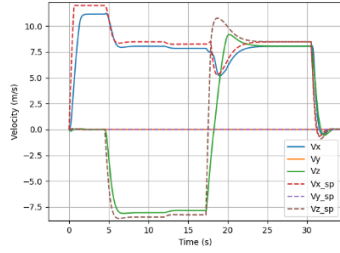


Speed vs Time
($v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}$)

Stage 2 – Segment 2 and 3: From Waypoint 1 to Waypoint 3

* *The speeds of the plots are inversely dimensioned (negative values when climbing).*

- The DJI F450 FAST is able to maintain maximum speed when climbing and descending for any given path angle. The greater the angle of the trajectory, the greater the position error, since the drone is not accurate enough to regain the trajectory precisely in a steep turn. Observe the green line in the position error graphs. The second error 'step' corresponding to the descent phase is greater than the error when climbing. This is because the drone turns $180 \text{ degrees} + 2 * \text{path angle degrees}$ in the top of climb waypoint.



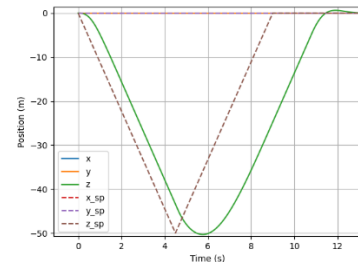
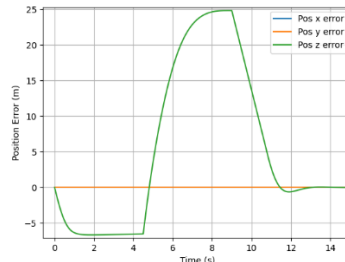
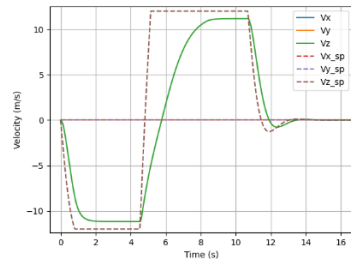
Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}; \text{distance} = 100\text{m per segment}; \text{inclination} = 45^{\circ})$

Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}; \text{distance} = 100\text{m per segment}; \text{inclination} = 60^{\circ})$

Speed and Position Error vs Time
 $(v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}; \text{distance} = 100\text{m per segment}; \text{inclination} = 80^{\circ})$

Stage 3 – Segment 4 and 5: From Waypoint 3 to Waypoint 4 to Waypoint 3

- The drone is able to reach the maximum speed of 11.1 m/s when climbing and descending vertically (90 degrees with respect to the horizontal plane). But due to the drone’s initial position error, the predicted position will be inaccurate. When reaching the top of climb waypoint, the drone will be delayed and the predicted position will be distorted, hence making such high position error. At smaller speeds, the drone has less position error.

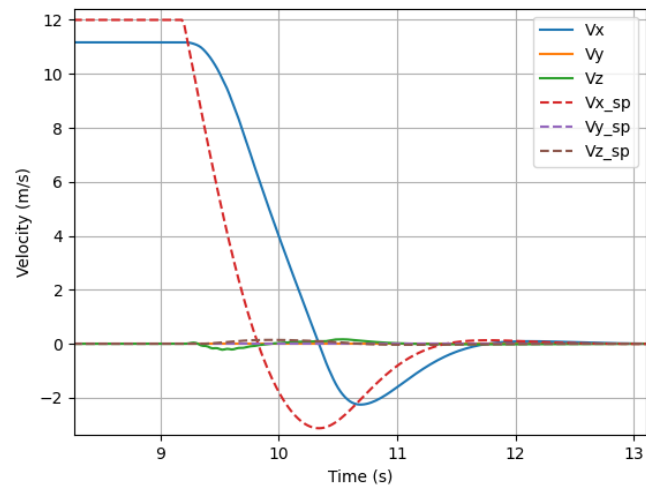


Speed and Position Error vs Time $(v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}; \text{distance} = 50\text{m})$

Stage 4 – Segment 6: From Waypoint 3 to Waypoint 5

- The drone decelerates from a stable maximum speed to 0 m/s in $t = 11.8\text{s} - 9.3\text{s} = 2.5\text{s}$. This gives a maximum braking deceleration of

$$a_{\text{max}} = \frac{-11.1 \frac{\text{m}}{\text{s}}}{2.5\text{s}} = -4.44 \frac{\text{m}}{\text{s}^2}$$



Speed vs Time ($v_{\text{desired}}^{\text{mean}} = 11.1 \frac{\text{m}}{\text{s}}$; $\text{distance} = 100\text{m}$)

APPENDIX D. DEG ARCHITECTURAL DESIGN

The architectural design of DEG is illustrated in [Table 0.1](#) and described in detail in [Table 0.2](#). Tables are sorted by hierarchy of DEG functions and the order in which they are invoked. The Python packages needed for this project are Numpy, Matplotlib, Pandas, PyDy and SimPy.

Table 0.1 Architectural Design of DEG.

Main Script

DEG-Encounter_generator.py

 Encounter_Distribution()

 parseFromJson()

 Parameters()

 load_params_from_file()

 CreateEncounterSetFolder()

Encounter Generation Loop

 CreateEncounterFolder()

 SampleNMAC()

Ownship Trajectory Generation

 sampleNMACSegment()

Segment Generation Loop

 sampleForwardSegment()

 sampleBackwardSegment()

 addLastBackwardSegment

Sort Segments

 positionNMACSegment()

 positionForwardSegment()

 positionBackwardSegment()

Generate Waypoints from Segments

 generateWaypoints()

Trajectory Generation

 generateTrajectory()

 Quadcopter()

 Trajectory()

 Control()

Simulation Loop

 quad_sim()

 update()

 desiredStates()

 controller()

Ownship Rotation Parameters

 NMACRotationParameters()

 setParametersFromEncounter()

 fillOwnshipFromIdx()

Intruder Trajectory Generation and Rotation Parameters

 (...)

 same procedure as for the ownship aircraft

 (...)

Rotate Intruder Path

```

adjustTrajectories()
Store Encounter
storeEncounterSummaryOutput()
Ownship Data
storeTrajectory()
storeFlowSequence()
storeCombinedTrajectory()
Intruder Data
storeTrajectory()
storeFlowSequence()
storeCombinedTrajectory()
End of Encounter Generation Loop
End of DEG

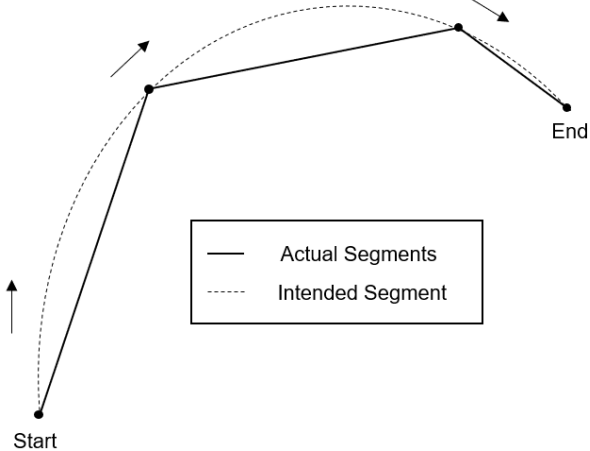
```

Table 0.2 DEG Component Descriptions.

Component	Description
DEG- Encounter_generator.py	This is the main script. It allows loading all the necessary variables and constants and executing all the subroutines of DEG, invoked following the hierarchy shown in Table 0.1.
Encounter_Distribution()	<i>Class.</i> Initializes the encounter descriptive parameters, distribution parameters and the aircraft class list (name, version, description, Aircraft_Class_List, Initial_Distribution, Transition_Distribution).
parseFromJson()	<i>Method.</i> Loads encounter probability distributions from <i>droneTestDistribution.json</i> for four different vehicles, two mission oriented and two cargo oriented.
Parameters()	<i>Class.</i> Initializes the top-level encounter parameters (generatedEncounters, totalNumberEncounters, probDistributions, destinationPath, encounterSetName)
load_params_from_file()	<i>Method.</i> Loads vehicle characteristics contained in <i>vehicleParams.json</i> .
CreateEncounterSetFolder()	<i>Method.</i> Creates generic output folder.
CreateEncounterFolder()	<i>Method.</i> Creates folder for each encounter and subfolders for each aircraft.
SampleNMAC()	<i>Method.</i> Sets initial conditions at CPA by sampling NMAC values (VMD, HMD, AppAngle, Altitude (Own,Intr), Speed (Own,Intr), ClassIdx (Own, Intr), Class (Own,Intr). These are obtained by Monte Carlo sampling (random stochastic sample) from a probability distribution contained in <i>droneTestDistribution.json</i> . Note that the Intruder's altitude at CPA is the ownship's altitude plus VMD.

sampleNMACSegment()	<p><i>Method.</i> Samples NMAC segment parameters by Monte Carlo sampling (random stochastic sample) from a probability distribution contained in <i>droneTestDistribution.json</i>:</p> <ul style="list-style-type: none"> - Gets a unitary form value (NMACFactor) ranging from 0 to 1. This will specify NMAC location as a percentage of total NMAC segment. - Determines the horizontal mode (1: constant, 2: turn, 3: hover). - Computes segment duration and speed variation (1: constant, 2: accelerate, 3: decelerate). If speed mode is constant or horizontal mode is hover then the segment duration is sampled from the distribution. - Gets initial and target (final) speeds of NMAC segment using same percentage provided by NMACFactor. If the target speed is greater than 0kts and the next segment horizontal mode is hovering, then it is calculated the time and distance required to stop having a deceleration factor of G/2. - Assigns heading variations if in a turn. The heading variation and the rate of turn are sampled from the distribution tables. - Determines vertical mode (1: level, 2: climb, 3: descent). If no level, computes altitude variation using NMACFactor. - Computes the target altitude, altitude variation and updates the segment duration depending on the vertical mode and the vehicle's performance. - Adds segment to segment list (segmentList).
sampleForwardSegment()	<p><i>Method.</i> Samples segment parameters by Monte Carlo sampling (random stochastic sample) from a probability distribution contained in <i>droneTestDistribution.json</i>:</p> <ul style="list-style-type: none"> - Determines the horizontal mode (1: constant, 2: turn, 3: hover). - Computes segment duration and speed variation (1: constant, 2: accelerate, 3: decelerate). If previous segment has null horizontal speed, then the next segment speed mode will be to accelerate. If speed mode is constant or horizontal mode is hover then the segment duration is sampled from the distribution.

	<ul style="list-style-type: none"> - Calculates speed variation and target speed for each segment according to vehicle performances and speed modes. If the target speed is greater than 0kts and the next segment horizontal mode is hovering, then it is calculated the time and distance required to stop having a deceleration factor of G/2. - Assigns heading variations if in a turn. The heading variation and the rate of turn are sampled from the distribution tables. - Determines the vertical mode (1: level, 2: climb, 3: descent). If level, altitude variation is null. - Computes the target altitude, altitude variation and updates the segment duration depending on the vertical mode and the vehicle's performance. - Adds segment to segment list (segmentList). <p>The initial altitude, speed and heading of forward segments correspond to the final (target) parameters of the previous segment.</p>
sampleBackwardSegment()	<p><i>Method.</i> The final (target) altitude, speed and heading of backward segments correspond to the initial parameters of the subsequent segment on the list. Backward segments are computed as in sampleNMACForwardSegment() but inserted as first elements of the list of segments.</p>
addLastBackwardSegment	<p><i>Method.</i> Creates the initial segment. The drone hovers then accelerates to reach the initial conditions of the first segment of the list (the last backward segment). The total duration of the initial segment is 5 seconds.</p>
positionNMACSegment()	<p><i>Method.</i> It calculates the initial and final positions of the CPA segment expressed in 4D cartesian coordinates.</p> <ul style="list-style-type: none"> - If straight segment, a 2D velocity vector is calculated based on the target speed and initial heading parameters. This vector along with the segment duration allows for the computation of the 3D spatial coordinates using a simple linear projection. - If turning, the segments are divided into subsegments of 45 degrees of heading change. Finally, it is calculated the total segment duration and target positions based on the subsegments.

	<p>- If hovering, the initial and target altitudes and segment durations are obtained from <code>sampleNMACSegment()</code>.</p> 
<code>positionForwardSegment()</code>	<p><i>Method.</i> Similar procedure as in <code>positionNMACSegment()</code> but the initial coordinates and time of the new segment correspond to the final position and time of the previous segment.</p>
<code>positionBackwardSegment()</code>	<p><i>Method.</i> Similar procedure as in <code>positionNMACSegment()</code> but the target coordinates and time of the new segment correspond to the initial position and time of the next segment.</p>
<code>generateWaypoints()</code>	<p><i>Method.</i> Fills a 4D array (<code>wpSequence</code>) based on the generated segments. This array contains the initial 3D cartesian coordinates as well as the cumulative time duration for each segment (sum of segment durations).</p>
<code>generateTrajectory()</code>	<p><i>Method.</i> This function creates the drone trajectory by interpolating between the generated waypoints. It uses a built-in complex multirotor model to derive the drone's equations of motions throughout the entire trajectory and update its state variables, simulating the flight of a real DJIF450 quadcopter. Refer to Section 4.1.4 to find a detailed explanation.</p>
<code>Quadcopter()</code>	<p><i>Class.</i> Initializes the drone parameters, calculates the initial motor command to stable hover at the origin, initializes the drone state variables and sets the integrator to solve the drone's equation of motion. These variables will be updated at every time step in the simulation.</p>
<code>Trajectory()</code>	<p><i>Class.</i> Initializes the trajectory parameters and the waypoint parameters. These variables will be updated at every time step in the simulation.</p>

Control()	<i>Class.</i> Initializes all the control variables, which through the flight controller (similar to the PX4 autopilot), will be updated in each time interval.
quad_sim()	<i>Method.</i> This function contains the main subroutines that will update the state variables of the drone, will calculate the next desired states and controller commands for each time step.
update()	<i>Method.</i> Updates quadrotor's state variables (position, quaternions, velocity, angular speed, motor angular velocity, acceleration and angular rate of change) for each time interval, solving the Ordinary Differential Equation (ODE).
desiredStates()	<i>Method.</i> Calculates desired state variables (position, velocity and acceleration) and state derivatives for the next iteration. The quadcopter states have a refresh rate of 50 Hz (20 ms).
controller()	<i>Method.</i> Executes cascaded controller (PID) to minimize the position error based on a predefined control type and generate the motor commands for the next iteration.
NMACRotationParameters()	<i>Class.</i> Initializes the rotation parameters at NMAC reference point (or CPA).
setParametersFromEncounter()	<i>Method.</i> Retrieves the rotation parameters (HMD, VMD and approach angle) at NMAC reference point from the generated trajectory.
fillOwnshipFromIdx()	<i>Method.</i> Computes the state variables (4D point, speed, altitude, vertical speed and heading) at the NMAC reference point within the generated trajectory.
adjustTrajectories()	<i>Method.</i> Rotates the intruder path to match with the desired encounter geometry at CPA (or NMAC point).
storeEncounterSummaryOutput()	<i>Method.</i> Store all encounter variables in Excel format.
storeTrajectory()	<i>Method.</i> Store desired waypoint sequence directly from the generated segments (prior to the trajectory generation) in Excel format.
storeFlowSequence()	<i>Method.</i> Store actual flown waypoint sequence in Excel format.
storeCombinedTrajectory()	<i>Method.</i> Store actual flown trajectory variables in Excel format.